opsawg Z. Li, Ed.
Internet-Draft R. Gu, Ed.
Intended status: Standards Track China Mobile
Expires: September 8, 2017 J. Dong
Huawei Technologies
March 7, 2017

# Export BGP community information in IP Flow Information Export (IPFIX)
draft-ietf-opsawg-ipfix-bgp-community-01

Abstract

This draft specifies an extension to the IPFIX information model defined in [RFC7012] to export the BGP community [RFC1997] information. Three information elements, bgpCommunity, bgpSourceCommunityList and bgpDestinationCommunityList, are introduced in this document to carry the BGP community information. bgpCommunity, containing exactly one BGP community value, is used to consist the list in bgpSourceCommunityList and bgpDestinationCommunityList, which are corresponding to a specific flow’s source IP and destination IP respectively.

Table of Contents



## 1. Introduction

IP Flow Information Export (IPFIX) [RFC7011] provides network administrators with traffic flow information using the information elements (IEs) defined in [IANA-IPFIX] registries. Based on the traffic flow information, network administrators know the amount and direction of the traffic in their network, then they can optimize their network when needed. For example, they can steer some flows from the congested links to the low utilized links.

[IANA-IPFIX] has already defined the following IEs for traffic flow information exporting in different grain: sourceIPv4Address, sourceIPv4Prefix, destinationIPv4Address, destinationIPv4Prefix, bgpSourceAsNumber, bgpDestinationAsNumber, bgpNextHopIPv4Address, etc. In some circumstances, however, especially when traffic engineering and optimization are used in the Tier 1 or Tier 2 operators' backbone networks, traffic flow information based on these IEs is not suitable. Flow information based on IP address or IP prefix is much more mediculous. On the contrary, flow information based on AS number is too coarse. BGP community [RFC1997], which describes a group of routes sharing some common properties, is preferablely used for fine granularity traffic engineering

[Community-TE] [RFC4384]. Unfortunately, [IANA-IPFIX] has no IE defined for BGP community information, yet.

Flow information based on BGP community can be collected by a mediator defined in [RFC6183]. Mediator is responsible for the correlation between flow information and BGP community. However no IEs are defined in [RFC6183] for exporting BGP community information in IPFIX. Furthermore, to correlate the BGP community with the flow information, mediator needs to learn BGP routes and lookup in the BGP routing table to get the matching entry for the specific flow. Neither BGP route learning nor routing table lookup is trivial for a mediator. Mediator is mainly introduced to release the performance requirement for the exporter [RFC5982]. In fact, to obtain the information for BGP related IEs that have already been defined, such as bgpSourceAsNumber, bgpDestinationAsNumber, and bgpNextHopIPv4Address, etc, exporter has to hold the up-to-date BGP routing table and look up in the BGP routing table. The exporter can get the community information in the same procedure. So, getting BGP community information adds no more requirement for exporter. Some vendors have already implemented this feature in their exporters using private IEs. So, exporter is RECOMMENDED to export the BGP community information in IPFIX directly, other than the mediator.

This draft specifies an extension to the IPFIX information model defined in [RFC7012] to export the BGP community information. Three IEs, bgpCommunity, bgpSourceCommunityList and bgpDestinationCommunityList, are introduced to complete this task. bgpCommunity contains one BGP community value. BgpSourceCommunityList consists of a list of bgpCommunity corresponding with the source IP address of a specific flow, and bgpDestinationCommunityList consists of a list of bgpCommunity corresponding with the destination IP address of a specific flow.

BgpCommunity, bgpSourceCommunityList and bgpDestinationCommunityList IEs are applicable for both IPv4 and IPv6 traffic. Both exporter and mediator can use these three IEs to export BGP community information in IPFIX.

## 2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

## 3. BGP Community Information Elements

In order to export BGP community information along with other flow information defined by IPFIX, we need to introduce three new IEs. One is bgpCommunity, which is used to identify that the value in this IE is BGP community [RFC1997]. The other two are bgpSourceCommunityList and bgpDestinationCommunityList. They both are basicList [RFC6313] of bgpCommunity. bgpSourceCommunityList and bgpDestinationCommunityList are used to export BGP community information corresponding to a specific flow’s source IP and destination IP respectively. Flow information based on BGP community can then be accumulated and analysed by the collector or other applications.

The details of these three new introduced IEs are illustrated below, including name, ID, type, semantics, description and units.

### 3.1. bgpCommunity

| | |
|---|---|
| ElementID | to be assigned by IANA, 458 is suggested |
| Name | bgpCommunity |
| Data Type | unsigned32 |
| Data Type Semantics | identifier |
| Description | BGP community as defined in [RFC1997] |
| Units | none |

Figure 1: bgpCommunity

### 3.2. bgpSourceCommunityList

| | |
|---|---|
| ElementID | to be assigned by IANA, 459 is suggested |
| Name | bgpSourceCommunityList |
| Data Type | basicList, as specified in [RFC6313] |
| Data Type Semantics | list |
| Description | zero or more BGP communities corresponding with source IP address of a specific flow |
| Units | none |

Figure 2: bgpSourceCommunityList

### 3.3. bgpDestinationCommunityList

| | |
|---|---|
| ElementID | to be assigned by IANA, 460 is suggested |
| Name | bgpDestinationCommunityList |
| Data Type | basicList, as specified in [RFC6313] |
| Data Type Semantics | list |
| Description | zero or more BGP communities corresponding with destination IP address of a specific flow |
| Units | none |

Figure 3: bgpDestinationCommunityList

## 4. Security Considerations

This document only defines three new IEs for IPFIX. So, this document itself does not directly introduce security issues. The same security considerations as for the IPFIX Protocol Specification [RFC7011] and Information Model [RFC7012] apply.

As the BGP community information is deducible by other means, there are no increased privacy concerns.

## 5. IANA Considerations

This draft specifies three new IPFIX IEs, bgpCommunity, bgpSourceCommunityList and bgpDestinationCommunityList, to export BGP community information along with other flow information.

The Element IDs for these three IEs are solicited to be assigned by IANA. The following table is for IANA's reference to put in each field in the registry.

| ElementID | Name | Data Type | Data Type Semantics |
|---|---|---|---|
| TBA1 | bgpCommunity | unsigned32 | identifier |
| TBA2 | bgpSourceCommunityList | basicList | list |
| TBA3 | bgpDestinationCommunityList | basicList | list |

| ElementID | Description | Units |
|---|---|---|
| TBA1 | BGP community | |
| TBA2 | zero or more BGP communities corresponding with source IP address of a specific flow | |
| TBA3 | zero or more BGP communities corresponding with destination IP address of a specific flow | |

| ElementID | Range | References | Requester | Revision | date |
|---|---|---|---|---|---|
| TBA1 | | RFC1997 | this draft | 0 | |
| TBA2 | | RFC6313,RFC1997 | this draft | 0 | |
| TBA3 | | RFC6313,RFC1997 | this draft | 0 | |

Figure 4

## 6. Acknowledgements

The authors would like to thank Benoit Claise and Paul Aitken for their comments and suggestions to promote this document. Appreciations are given to Tianran Zhou, Jeffrey Haas, Ignas Bagdonas, Stewart Bryant, Paolo Lucente, Job Snijders, Jared Mauch, etc, for their discussion in the mail list.

## 7. References

### 7.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC6313] Claise, B., Dhandapani, G., Aitken, P., and S. Yates, "Export of Structured Data in IP Flow Information Export (IPFIX)", RFC 6313, DOI 10.17487/RFC6313, July 2011, <http://www.rfc-editor.org/info/rfc6313>.

[RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information", STD 77, RFC 7011, DOI 10.17487/RFC7011, September 2013, <http://www.rfc-editor.org/info/rfc7011>.

[RFC7012] Claise, B., Ed. and B. Trammell, Ed., "Information Model for IP Flow Information Export (IPFIX)", RFC 7012, DOI 10.17487/RFC7012, September 2013, <http://www.rfc-editor.org/info/rfc7012>.

### 7.2. Informative References

[Community-TE] Shao, W., Devienne, F., Iannone, L., and JL. Rougier, "On the use of BGP communities for fine-grained inbound traffic engineering", Computer Science 27392(1):476-487, November 2015.

[IANA-IPFIX] "IP Flow Information Export (IPFIX) Entities", <http://www.iana.org/assignments/ipfix/>.

[RFC1997] Chandra, R., Traina, P., and T. Li, "BGP Communities Attribute", RFC 1997, DOI 10.17487/RFC1997, August 1996, <http://www.rfc-editor.org/info/rfc1997>.

[RFC4384] Meyer, D., "BGP Communities for Data Collection", BCP 114, RFC 4384, DOI 10.17487/RFC4384, February 2006, <http://www.rfc-editor.org/info/rfc4384>.

[RFC5982] Kobayashi, A., Ed. and B. Claise, Ed., "IP Flow Information Export (IPFIX) Mediation: Problem Statement", RFC 5982, DOI 10.17487/RFC5982, August 2010, <http://www.rfc-editor.org/info/rfc5982>.

[RFC6183] Kobayashi, A., Claise, B., Muenz, G., and K. Ishibashi, "IP Flow Information Export (IPFIX) Mediation: Framework", RFC 6183, DOI 10.17487/RFC6183, April 2011, <http://www.rfc-editor.org/info/rfc6183>.

## Appendix A. Application Example

In this section, we give an example to show the encoding format for the three new introduced IEs.

Flow information including BGP communities is shown in the below table. Suppose we want all the fields to be reported by IPFIX.

| Source ip | Destination ip | Source BGP community | Destination BGP community |
|---|---|---|---|
| 1.1.1.1 | 2.2.2.2 | 1:1001,1:1002,8:1001 | 2:1002,8:1001 |
| 3.3.3.3 | 4.4.4.4 | 3:1001,3:1002,8:1001 | 4:1001,8:1001 |

Figure 5: Flow information including BGP communities

### A.1. Template Record

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          SET ID = 2           |       Length = 24             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Template ID = 256        |        Field Count = 4        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|    SourceIPv4Address = 8    |        Field length = 4       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0| DestinationIPv4Address = 12 |        Field length = 4       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0| bgpSourceCommunityList = 459|      Field length = 0xFFFF    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0| bgpDestinationCommunityList |      Field length = 0xFFFF    |
| |         = 460               |                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 6: Template Record Encoding Format

In this example, the Template ID is 256, which will be used in the data record. The field length for bgpSourceCommunityList and bgpDestinationCommunityList is 0xFFFF, which means the length of this IE is variable, the actual length of this IE is indicated by the list length field in the basic list format as per [RFC6313].

## A.2. Data Set

The data set is represented as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         SET ID = 256          |           Length = 92         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  SourceIPv4Address = 1.1.1.1                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               DestinationIPv4Address = 2.2.2.2                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      255      |         List length = 17      |semantic=allof |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      bgpCommunity = 458       |          Field Len = 4        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        BGP Source Community Value 1 = 1:1001                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        BGP Source Community Value 2 = 1:1002                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        BGP Source Community Value 3 = 8:1001                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
|     255       |         List length = 13       |semantic =allof|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      bgpCommunity = 458       |          Field Len = 4        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         BGP Destination Community Value 1 = 2:1002            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         BGP Destination Community Value 2 = 8:1001            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  SourceIPv4Address = 3.3.3.3                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               DestinationIPv4Address = 4.4.4.4                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     255       |         List length = 17       |semantic =allof|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      bgpCommunity = 458       |          Field Len = 4        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        BGP Source Community Value 1  = 3:1001                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        BGP Source Community Value 2  = 3:1002                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        BGP Source Community Value 3  = 8:1001                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     255       |         List length = 13       |semantic =allof|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      bgpCommunity = 458       |          Field Len = 4        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         BGP Destination Community Value 1 = 4:1001            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         BGP Destination Community Value 2 = 8:1001            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 7: Data Set Encoding Format

## Authors' Addresses

Zhenqiang Li (editor)
China Mobile
32 Xuanwumen West Ave, Xicheng District
Beijing  100053
China

Email: lizhenqiang@chinamobile.com

Rong Gu (editor)
China Mobile
32 Xuanwumen West Ave, Xicheng District
Beijing  100053
China

Email: gurong_cmcc@outlook.com

Jie Dong
Huawei Technologies
Huawei Campus, No. 156 Beiqing Rd.
Beijing  100095
China

Email: jie.dong@huawei.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: November 16, 2017

E. Lear
Cisco Systems
R. Droms

D. Romascanu

May 15, 2017

# Manufacturer Usage Description Specification
draft-ietf-opsawg-mud-06

## Abstract

This memo specifies a component-based architecture for manufacturer usage descriptions (MUD). This includes two YANG modules, IPv4 and IPv6 DHCP options, an LLDP TLV, a URL suffix specification, an X.509 certificate extension and a means to sign and verify the descriptions.

## Status of This Memo



## Copyright Notice

Table of Contents

## 1. Introduction

The Internet has largely been constructed on general purpose computers; those devices that may be used for a purpose that is specified by those who buy the device. [RFC1984] presumed that an end device would be most capable of protecting itself. This made sense when the typical device was a workstation or a mainframe, and it continues to make sense for general purpose computing devices today, including laptops, smart phones, and tablets.

[RFC7452] discusses design patterns for, and poses questions about, smart objects. Let us then posit a group of objects that are specifically NOT general purpose computers. These devices therefore have a purpose to their use. By definition, therefore, all other purposes are NOT intended. The combination of these two statements can be restated as a manufacturer usage description (MUD) that can be applied at various points within a network. Although this memo may seem to stress access requirements, usage intent also consists of quality of service needs a device may have.

We use the notion of "manufacturer" loosely in this context, to simply mean the entity or organization that will state how a device is intended to be used. In the context of a lightbulb, this might indeed be the lightbulb manufacturer. In the context of a smarter device that has a built in Linux stack, it might be integrator of that device. The key points are that the device itself is expected to serve a limited purpose, and that there may exist an organization in the supply chain of that device that will take responsibility for informing the network about that purpose.

The intent of MUD is to therefore solve for the following problems:

- o Substantially reduce the threat surface on a device entering a network to those communications intended by the manufacturer.

- o Provide for a means to scale network policies to the ever-increasing number types of devices in the network.

- o Provide a means to address at least some vulnerabilities in a way that is faster than it might take to update systems. This will be particularly true for systems that are no longer supported by their manufacturer.

- o Keep the cost of implementation of such a system to the bare minimum.

MUD therefore consists of three architectural building blocks: - A classifier that a device emits that can be used to locate a description; - The description itself, including how it is interpreted, and; - A means to retrieve the description.

In this specification we specify each of these building blocks and how they are intended to be used together. However, they may also be used separately, independent of this specification by enterprise networks for their own purposes.

### 1.1. What MUD doesn’t do

General computing systems will benefit very little from MUD, as their manufacturers cannot envision a specific communication pattern to describe. In addition, even those devices that have a single or small number of uses might have very broad communication patterns. MUD is not for them either.

No matter how good a MUD-enabled network is, it will never replace the need for manufacturers to patch vulnerabilities. It may, however, provide network administrators with some additional protection when those vulnerabilities exist.

### 1.2. A Simple Example

A light bulb is intended to light a room. It may be remotely controlled through the network; and it may make use of a rendezvous service of some form that an app on smart phone accesses. What we can say about that light bulb, then, is that all other network access is unwanted. It will not contact a news service, nor speak to the refrigerator, and it has no need of a printer or other devices. It has no Facebook friends. Therefore, an access list applied to it that states that it will only connect to the single rendezvous service will not impede the light bulb in performing its function, while at the same time allowing the network to provide both it and other devices an additional layer of protection.

## 1.3. Determining Intended Use

The notion of intended use is in itself not new. Network administrators apply access lists every day to allow for only such use. This notion of white listing was well described by Chapman and Zwicky in [FW95]. Programmatically profiling systems have existed for years as well. These systems make use of heuristics that take at least some time to assert what a system is.

A system could just as easily tell the network what sort of protection it requires without going into what sort of system it is. This would, in effect, be the converse of [RFC7488]. In seeking a general purpose solution, however, we assume that a device has so few capabilities that it will implement the least necessary capabilities to function properly. This is a basic economic constraint. Unless the network would refuse access to such a device, its developers would have no reason to implement such an approach. To date, such an assertion has held true.

## 1.4. Finding A Policy: The MUD URL

Our work begins, therefore, with the device emitting a Universal Resource Locator (URL) [RFC3986]. This URL may serves both to classify the device type and to provide a means to locate a policy file.

In this memo three means are defined to emit the MUD URL. One is a DHCP option[RFC2131],[RFC3315] that the DHCP client uses to inform the DHCP server. The DHCP server may take further actions, such as retrieve the URL or otherwise pass it along to network management system or controller. The other method defined is an X.509 constraint. The IEEE has developed [IEEE8021AR] that provides a certificate-based approach to communicate device characteristics, which itself relies on [RFC5280]. The MUD URL extension is non-critical, as required by IEEE 802.1AR. Various means may be used to communicate that certificate, including Tunnel Extensible Authentication Protocol (TEAP) [RFC7170]. Finally, a Link Layer Discovery Protocol (LLDP) frame is defined [IEEE8021AB].

It is possible that there may be other means for a MUD URL to be learned by a network. For instance, if a device has a serial number, it may be possible for the MUD controller to perform a lookup of the device, if it has some knowledge as to who the device manufacturer is, and what its MUD file server is. Such mechanisms are not described in this memo, but are possible.

### 1.5. Types of Policies

When the MUD URL is resolved, the MUD controller retrieves a file that describes what sort of communications a device is designed to have. The manufacturer may specify either specific hosts for cloud based services or certain classes for access within an operational network. An example of a class might be "devices of a specified manufacturer type", where the manufacturer type itself is indicated simply by the authority component (e.g, the domain name) of the MUD URL. Another example might to allow or disallow local access. Just like other policies, these may be combined. For example:

```
Allow access to devices of the same manufacturer
Allow access to and from controllers who need to speak COAP
Allow access to local DNS/DHCP
Deny all other access
```

To add a bit more depth that should not be a stretch of anyone's imagination, one could also make use of port-based access lists. Thus a printer might have a description that states:

```
Allow access for port IPP or port LPD
Allow local access for port HTTP
Deny all other access
```

In this way anyone can print to the printer, but local access would be required for the management interface.

The files that are retrieved are intended to be closely aligned to existing network architectures so that they are easy to deploy. We make use of YANG [RFC6020] because of the time and effort spent to develop accurate and adequate models for use by network devices. JSON is used as a serialization for compactness and readability, relative to XML.

The YANG modules specified here are extensions of [I-D.ietf-netmod-acl-model]. The extensions to this model allow for a manufacturer to express classes of systems that a manufacturer would find necessary for the proper function of the device. Two modules are specified. The first module specifies a means for domain names to be used in ACLs so that devices that have their controllers in the cloud may be appropriately authorized with domain names, where the mapping of those names to addresses may rapidly change.

The other module abstracts away IP addresses into certain classes that are instantiated into actual IP addresses through local processing. Through these classes, manufacturers can specify how the device is designed to communicate, so that network elements can be configured by local systems that have local topological knowledge. That is, the deployment populates the classes that the manufacturer specifies. The abstractions are as follows:

Manufacturer: A device made by a particular manufacturer, as identified by the authority component of its MUD-URL

same-manufacturer: Devices that have the same authority component of their MUD-URL.

Controller: A device that the local network administrator admits to the particular class.

my-controller: A class associated with the MUD-URL of a device that the administrator admits.

The "manufacturer" classes can be easily specified by the manufacturer, whereas controller classes are initially envisioned to be specified by the administrator.

Because manufacturers do not know who will be using their devices, it is important for functionality referenced in usage descriptions to be relatively ubiquitous, and therefore, mature. Therefore, only a a limited subset YANG-based configuration of is permitted in a MUD file.

## 1.6. Terminology

MUD: manufacturer usage description.

MUD file: a file containing YANG-based JSON that describes a recommended behavior.

MUD file server: a web server that hosts a MUD file.

MUD controller: the system that requests and receives the MUD file from the MUD server. After it has processed a MUD file it may direct changes to relevant network elements.

MUD URL: a URL that can be used by the MUD controller to receive the MUD file.

Thing: the end device that emits a MUD URL.

Manufacturer: the entity that configures the Thing to emit the MUD URL and the one who asserts a recommendation in a MUD file. The manufacturer might not always be the entity that constructs a device. It could, for instance, be a systems integrator, or even a component provider.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

## 1.7. The Manufacturer Usage Description Architecture

With these components laid out we now have the basis for an archicture. This leads us to ASCII art.

```
 .........................................
 .                      ____________     .          _____________
 .                     |            |    .         |             |
 .                     |    MUD     |----->get URL->|    MUD      |
 .                     | Controller |    .(https)   | File Server |
 .  End system network |____________|<--MUD file<--<|_____________|
 .                             .         .
 .                             .         .
 . _______                 _________     .
 .|       |  (dhcp et al) | router  |    .
 .| Thing |---->MUD URL-->|   or    |    .
 .|_______|               | switch  |    .
 .                        |_________|    .
 .........................................
```

Figure 1: MUD Architecture

In the above diagram, the switch or router collects MUD URLs and forwards them to the network management system for processing. This happens in different ways, depending on how the URI is communicated. For instance, in the case of DHCP, the DHCP server might receive the URI and then process it. In the case of IEEE 802.1X, the switch would carry the URI via a certificate to the authentication server via EAP over Radius[RFC3748], which would then process it. One method to do this is TEAP, described in [RFC7170]. The certificate extension is described below.

The information returned by the web site is valid for the duration of the device's connection, or as specified in the description. Thus if the device is mobile, when it moves on, any configuration in the switch is removed. Similarly, from time to time the description may be refreshed, based on new capabilities or communication patterns or vulnerabilities.

The web site is typically run by or on behalf of the manufacturer. Its domain name is that of the authority found in the MUD URL. For legacy cases where Things cannot emit a URL, if the switch is able to determine the appropriate URI, it may proxy it, the trivial cases being a map between some registered device or port and a URL.

### 1.8. Order of operations

As mentioned above, MUD contains architectural building blocks, and so order of operation may vary. However, here is one clear intended example:

1. Device emits URL.

2. That URL is forwarded to a MUD controller by the nearest switch (how this happens depends on the way in which the MUD URL is emitted).

3. The MUD controller retrieves the MUD file from the MUD file server, assuming it doesn't already have a copy. It may test the URL against a reputation service, and it may test any hosts within the file against reputation services, as it deems fit.

4. The MUD controller may query the administrator for permission to add the device and associated policy. If the device is known or the device type is known, it may skip this step.

5. The MUD controller instantiates local configuration based on the abstractions defined in this document.

6. The MUD controller configures the switch nearest the device. Other systems may be configured as well.

7. When the device disconnects, policy is removed.

## 2. The MUD Model and Semantic Meaning

A MUD file consists of JSON based on a YANG model. For purposes of MUD, the elements that can be modified are access lists as augmented by this model. The MUD file is limited to the serialization of a small number of YANG schema, including the models specified in the following documents:

- [I-D.ietf-netmod-acl-model]

- [RFC6991]

Publishers of MUD files MUST NOT include other elements except as described in Section 3.6, and MUST only contain information relevant to the device being described. Devices parsing MUD files MUST cease processing if they find other elements.

This module is structured into three parts. The first container holds information that is relevant to retrieval and validity of the MUD file itself. The second container augments the access list to indicate direction the ACL is to be applied. The final container augments the matching container of the ACL model to add several elements that are relevant to the MUD URL, or other otherwise abstracted for use within a local environment.

Simplified graphical representation of the data models are used in this document. The meaning of the symbols in these diagrams is defined in [I-D.ietf-netmod-rfc6087bis].

```
    +--rw metainfo
       +--rw last-update?      yang:date-and-time
       +--rw cache-validity?   uint8
       +--rw masa-server?      inet:uri
       +--rw is-supported?     boolean
       +--rw systeminfo?       inet:uri
       +--rw extensions*       string
  augment /acl:access-lists/acl:acl:
    +--rw packet-direction?   direction
  augment /acl:access-lists/acl:acl/acl:access-list-entries/
  acl:ace/acl:matches:
    +--rw manufacturer?          inet:host
    +--rw same-manufacturer?     empty
    +--rw model?                 string
    +--rw local-networks?        empty
    +--rw controller?            inet:uri
    +--rw my-controller?         empty
    +--rw direction-initiated?   direction
```

## 3. Element Definitions

The following elements are defined.

### 3.1. last-update

This is a date-and-time value of when the MUD file was generated. This is akin to a version number. Its form is taken from [RFC6991] which, for those keeping score, turn was taken from Section 5.6 of [RFC3339], which was taken from [ISO.8601.1988].

### 3.2. cache-validity

This uint8 is the period of time in hours that a network management station MUST wait since its last retrieval before checking for an update. It is RECOMMENDED that this value be no less than 24 and MUST NOT be more than 168 for any device that is supported.

### 3.3. masa-server

This optional element refers to the URL that should be used to resolve the location any MASA service, as specified in [I-D.ietf-anima-bootstrapping-keyinfra].

### 3.4. is-supported

This boolean is an indication from the manufacturer to the network administrator as to whether or not the device is supported. In this context a device is said to be supported if the manufacturer might issue an update to the device or if the manufacturer might update the MUD file.

### 3.5. systeminfo

This is a URL that points to a description of the device to be connected. The intent is for administrators to be able to read about what the device is the first time the MUD-URL is used.

### 3.6. extensions

This optional leaf-list names MUD extensions that are used in the MUD file. Note that NO MUD extensions may be used in a MUD file prior to the extensions being declared. Implementations MUST ignore any elements in this file that they do not understand.

### 3.7. packet-direction

[I-D.ietf-netmod-acl-model] describes access-lists but does not attempt to indicate where they are applied as that is handled elsewhere in a configuration. However, in this case, a MUD file must be explicit in describing the communication pattern of a device, and that includes indicating what is to be permitted or denied in either direction of communication. This element takes a single value of either "to-device" or "from-device", based on a typedef "direction".

### 3.8. manufacturer

This element consists of a hostname that would be matched against the authority component of another device's MUD URL.

### 3.9. same-manufacturer

This is an equivalent for when the manufacturer element is used to indicate the authority that is found in another device's MUD URL matches that of the authority found in this device's MUD URL.

### 3.10. model

This string matches the one and only segment following the authority component of the MUD URL. It refers to a model that is unique within the context of the authority. It may also include product version information. Thus how this field is constructed is entirely a local matter for the manufacturer.

### 3.11. local-networks

This null-valued element expands to include local networks. Its default expansion is that packets must not traverse toward a default route that is received from the router. However, administrators may expand the expression as is appropriate in their deployments.

### 3.12. controller

This URI specifies a value that a controller will register with the mud controller. The element then is expanded to the set of hosts that are so registered. This element may also be a URN. In this case, the URN describes a well known service, such as DNS or NTP.

Great care should be used when invoking the controller class. For one thing, it requires some understanding by the administrator as to when it is appropriate. Classes that are standardized may make it possible to code in certain intelligence. Nonstandard classes may require substantially more care. Pre-registration in such classes by controllers with the MUD server is encouraged. The mechanism to do that is beyond the scope of this work.

### 3.13. my-controller

This null-valued element establishes a class of controllers that are intended to control the device associated with the MUD file being referenced.

### 3.14. direction-initiated

When applied this matches packets when the flow was initiated in the corresponding direction. [RFC6092] specifies IPv6 guidance best practices. While that document is scoped specifically to IPv6, its contents are applicable for IPv4 as well. When this flag is set, and the system has no reason to believe a flow has been initiated it MUST drop the packet. This match SHOULD be applied with specific transport parameters, such as protocol.

## 4. Processing of the MUD file

To keep things relatively simple in addition to whatever definitions exist, we also apply two additional default behaviors:

- Anything not explicitly permitted is denied.
- Local DNS and NTP are, by default, permitted to and from the device.

An explicit description of the defaults can be found in Appendix B.

## 5. What does a MUD URL look like?

To begin with, MUD takes full advantage of both the https: scheme and the use of .well-known. HTTPS is important in this case because a man in the middle attack could otherwise harm the operation of a class of devices. .well-known is used because we wish to add additional structure to the URL. And so the URL appears as follows:

```
mud-url   = "https://" authority  "/.well-known/mud/" mud-rev
            "/" model ( "?" extras )
            ; authority is from RFC3986
mud-rev   = "v1"
model     = segment  ; from RFC3986
extras    = query    ; from RFC3986
```

mud-rev signifies the version of the manufacturer usage description file. This memo specifies "v1" of that file. Later versions may permit additional schemas or modify the format. In order to provide for the broadest compatibility for the various transmission mechanisms, the length of the URL for v1 MUST NOT exceed 255 octets.

"model" represents a device model as the manufacturer wishes to represent it. It could be a brand name or something more specific. It also may provide a means to indicate what version the product is. Specifically if it has been updated in the field, this is the place where evidence of that update would appear. The field should be changed when the intended communication patterns of a device change. While from a controller standpoint, only comparison and matching operations are safe, it is envisioned that updates will require some administrative review. Processing of this URL occurs as specified in [RFC2818] and [RFC3986].

## 6. The MUD YANG Model

```
<CODE BEGINS>file "ietf-mud@2017-04-18.yang"
module ietf-mud {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-mud";
  prefix "ietf-mud";

  import ietf-access-control-list {
    prefix "acl";
  }

  import ietf-yang-types {
    prefix "yang";
  }

  import ietf-inet-types {
    prefix "inet";
  }

  organization
    "IETF OPSAWG (Ops Area) Working Group";

  contact
       "WG Web: http://tools.ietf.org/wg/opsawg/
       WG List: opsawg@ietf.org
       Author: Eliot Lear
       lear@cisco.com
       Author: Ralph Droms
       rdroms@gmail.com
       Author: Dan Romascanu
       dromasca@gmail.com

    ";

  description
    "This YANG module defines a component that augments the
     IETF description of an access list.  This specific module
     focuses on additional filters that include local, model,
     and same-manufacturer.
```

```
    Copyright (c) 2016,2017 IETF Trust and the persons
    identified as the document authors.  All rights reserved.
    Redistribution and use in source and binary forms, with or
    without modification, is permitted pursuant to, and subject
    to the license terms contained in, the Simplified BSD
    License set forth in Section 4.c of the IETF Trust's Legal
    Provisions Relating to IETF Documents
    (http://trustee.ietf.org/license-info).
    This version of this YANG module is part of RFC XXXX; see
    the RFC itself for full legal notices.";

  revision "2017-04-18"  {
    description "Base version of MUD extensions to ACL model";
    reference "RFC XXXX: Manufacturer Usage Description
               Specification";
  }

  typedef direction {
     type enumeration {
         enum to-device {
           description "packets or flows destined to the target
                        device";
         }
         enum from-device {
           description "packets or flows destined from
                        the target device";
           }
         }
     description "Which way are we talking about?";
  }

  container metainfo  {

    description "Information about when support end(ed), and
                 when to refresh";

    leaf last-update {
      type yang:date-and-time;
      description "This is intended to be when
                   the MUD file was generated.";
    }

    leaf cache-validity {
      type uint8 {
        range "1..168";
      }
      description "The information retrieved from the MUD server is
                 valid for these many hours, after which it should
```

```
                 be refreshed.";
    }

    leaf masa-server {
      type inet:uri;
      description "The URI of the MASA server that network
      elements should forward requests to for this device.";
    }

    leaf is-supported {
      type boolean;
      description "The element is currently supported
                   by the manufacturer.";
    }
    leaf systeminfo {
      type inet:uri;
      description "A reference to a description of this device";
    }

    leaf-list extensions {
      type string;
      description "A list of extension names that are used in this MUD
                   file.  Each name is registered with the IANA and
                   described in an RFC.";
   }
  }

  augment "/acl:access-lists/acl:acl"   {
    description "add inbound or outbound.  Normally access lists
                 are applied in an inbound or outbound direction
                 separately from their definition.  This is not
                 possible with MUD.";
    leaf packet-direction
    {
      type direction;
      description "inbound or outbound ACL.";
    }
  }


  augment "/acl:access-lists/acl:acl/" +
     "acl:access-list-entries/acl:ace/" +
     "acl:matches" {
    description "adding abstractions to avoid need of IP addresses";

    leaf manufacturer {
      type inet:host;
      description "authority component of the manufacturer URI";
```

```
    }

    leaf same-manufacturer {
      type empty;
      description "expand to ACEs for each device
                   with the same origin";
    }

    leaf model {
        type string;
        description "specific model (including version) for a
                     given manufacturer";
    }

    leaf local-networks {
      type empty;
      description "this string is used to indicate networks
                   considered local in a given environment.";
    }

    leaf controller {
      type inet:uri;
      description "expands to one or more controllers for a
                   given service that is codified by inet:uri.";
    }

    leaf my-controller {
      type empty;
      description "This element indicates that the network should manage
                   a class of devices related to this MUD-URL that are
                   intended to control this device.";
    }
    leaf direction-initiated {
      type direction;
      description "which direction a flow was initiated";
    }
  }
}

<CODE ENDS>
```

## 7. The Domain Name Extension to the ACL Model

This module specifies an extension to IETF-ACL model such that domain names may be referenced by augmenting the "matches" element. Different implementations may deploy differing methods to maintain the mapping between IP address and domain name, if indeed any are needed. However, the intent is that resources that are referred to using a name should be authorized (or not) within an access list.

The structure of the change is as follows:

```
augment
/acl:access-lists/acl:acl/acl:access-list-entries
   /acl:ace/acl:matches/acl:ace-type/acl:ace-ip:
   +--rw src-dnsname?        inet:host
   +--rw dst-dnsname?        inet:host
```

The choice of this particular point in the access-list model is based on the assumption that we are in some way referring to IP-related resources, as that is what the DNS returns. A domain name in our context is defined in [RFC6991].

The following elements are defined.

## 7.1. source-dnsname

The argument corresponds to a domain name of a source as specified by inet:host. Depending on how the model is used, it may or may not be resolved, as required by the implementation and circumstances.

## 7.2. destination-dnsname

The argument corresponds to a domain name of a destination as specified by inet:host. Depending on how the model is used, it may or may not be resolved, as required by the implementation and circumstances.

## 7.3. The ietf-acldns Model

```
<CODE BEGINS>file "ietf-acldns@2016-07-20.yang"

module ietf-acldns {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-acldns";
  prefix "ietf-acldns";

  import ietf-access-control-list {
    prefix "acl";
  }

  import ietf-inet-types {
    prefix "inet";
  }
```

```
     organization
       "IETF OPSAWG (Ops Area) Working Group";

     contact
          "WG Web: http://tools.ietf.org/wg/opsawg/
          WG List: opsawg@ietf.org
          Author: Eliot Lear
          lear@cisco.com
          Author: Ralph Droms
          rdroms@gmail.com
          Author: Dan Romascanu
          dromasca@gmail.com
       ";


     description
       "This YANG module defines a component that augments the
        IETF description of an access list to allow dns names
        as matching criteria.";

     revision "2016-07-20" {
       description "Base version of dnsname extension of ACL model";
       reference "RFC XXXX: Manufacturer Usage Description
                  Specification";
     }

     augment "/acl:access-lists/acl:acl/" +
        "acl:access-list-entries/acl:ace/" +
        "acl:matches/acl:ace-type/acl:ace-ip" {
       description "adding domain names to matching";

       leaf src-dnsname {
         type inet:host;
         description "domain name to be matched against";
       }
       leaf dst-dnsname {
         type inet:host;
         description "domain name to be matched against";
       }
     }
   }

   <CODE ENDS>
```

## 8.  MUD File Example

This example contains two access lists that are intended to provide outbound access to a cloud service on TCP port 443.


```
{
   "ietf-mud:metainfo": {
      "last-update": "2016-05-18T20:00:50Z",
      "cache-validity": 168
      },
   "ietf-access-control-list:access-lists":  {
     "acl": [ {
      "acl-name": "inbound-stuff",
      "acl-type" : "ipv4-acl",
      "ietf-mud:packet-direction" : "to-device",
      "access-list-entries": {
         "ace": [
            {
               "rule-name": "access-cloud",
               "matches": {
                 "ietf-acldns:src-dnsname":
                     "lighting-system.example.com",
                  "protocol" : 6,
                  "source-port-range" : {
                     "lower-port" : 443,
                     "upper-port" : 443
                     }
                  },
               "actions" : {
                 "permit" : [null]
                 }
              }
            ]
           }
         },
         {
      "acl-name": "outbound-stuff",
      "acl-type" : "ipv4-acl",
      "ietf-mud:packet-direction" : "from-device",
      "access-list-entries": {
         "ace": [
            {
               "rule-name": "access-cloud",
               "matches": {
                  "ietf-acldns:dst-dnsname":
                       "lighting-system.example.com",
                   "protocol" : 6,
                   "destination-port-range" : {
```

```
                        "lower-port" : 443,
                        "upper-port" : 443
                       }
                   },
                  "actions" : {
                      "permit" : [null]
                    }
                 }
               ]
             }
           }
          ]
       }
   }
```


## 9. The MUD URL DHCP Option

The IPv4 MUD URL client option has the following format:

```
+------+-----+------------------------------
| code | len |  MUD URL
+------+-----+------------------------------
```

Code OPTION_MUD_URL_V4 (161) is assigned by IANA. len is a single octet that indicates the length of the URL in octets. MUD URL is a URL. MUD URLs MUST NOT exceed 255 octets.

The IPv6 MUD URL client option has the following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         OPTION_MUD_URL_V6     |        option-length          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             MUD URL                           |
|                               ...                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

OPTION_MUD_URL_V6 (112; assigned by IANA).

option-length contains the length of the URL in octets.

The intent of this option is to provide both a new device classifier to the network as well as some recommended configuration to the routers that implement policy. However, it is entirely the purview of the network system as managed by the network administrator to decide what to do with this information. The key function of this option is simply to identify the type of device to the network in a structured way such that the policy can be easily found with existing toolsets.

## 9.1. Client Behavior

A DHCPv4 client MAY emit a DHCPv4 option and a DHCPv6 client MAY emit DHCPv6 option. These options are singletons, as specified in [RFC7227]. Because clients are intended to have at most one MUD URL associated with them, they may emit at most one MUD URL option via DHCPv4 and one MUD URL option via DHCPv6. In the case where both v4 and v6 DHCP options are emitted, the same URL MUST be used.

Clients SHOULD log or otherwise report improper acknowledgments from servers, but they MUST NOT modify their MUD URL configuration based on a server’s response. The server’s response is only an acknowledgment that the server has processed the option, and promises no specific network behavior to the client. In particular, it may not be possible for the server to retrieve the file associated with the MUD URL, or the local network administration may not wish to use the usage description. Neither of these situations should be considered in any way exceptional.

## 9.2. Server Behavior

A DHCP server may ignore these options or take action based on receipt of these options. If a server successfully parses the option and the URL, it MUST return the option with length field set to zero and a corresponding null URL field as an acknowledgment. Even in this circumstance, no specific network behavior is guaranteed. When a server consumes this option, it will either forward the URL and relevant client information to a network management system (such as the giaddr), or it will retrieve the usage description by resolving the URL.

DHCP servers may implement MUD functionality themselves or they may pass along appropriate information to a network management system or MUD controller. A DHCP server that does process the MUD URL MUST adhere to the process specified in [RFC2818] and [RFC5280] to validate the TLS certificate of the web server hosting the MUD file. Those servers will retrieve the file, process it, create and install the necessary configuration on the relevant network element. Servers SHOULD monitor the gateway for state changes on a given interface. A DHCP server that does not provide MUD functionality and has forwarded a MUD URL to a MUD controller MUST notify the MUD controller of any corresponding change to the DHCP state of the client (such as expiration or explicit release of a network address lease).

## 9.3. Relay Requirements

There are no additional requirements for relays.

# 10. The Manufacturer Usage Description (MUD) URL X.509 Extension

This section defines an X.509 non-critical certificate extension that contains a single Uniform Resource Identifier (URI) that points to an on-line Manufacturer Usage Description concerning the certificate subject. URI must be represented as described in Section 7.4 of [RFC5280].

Any Internationalized Resource Identifiers (IRIs) MUST be mapped to URIs as specified in Section 3.1 of [RFC3987] before they are placed in the certificate extension.

The semantics of the URI are defined Section 5 of this document.

The choice of id-pe is based on guidance found in Section 4.2.2 of [RFC5280]:

> These extensions may be used to direct applications to on-line information about the issuer or the subject.

The MUD URL is precisely that: online information about the particular subject.

The new extension is identified as follows:

```
<CODE BEGINS>

  MUDURLExtnModule-2016 { iso(1) identified-organization(3) dod(6)
               internet(1) security(5) mechanisms(5) pkix(7)
               id-mod(0) id-mod-mudURLExtn2016(88) }

  DEFINITIONS IMPLICIT TAGS ::= BEGIN

  -- EXPORTS ALL --

  IMPORTS
    EXTENSION
    FROM PKIX-CommonTypes-2009
            { iso(1) identified-organization(3) dod(6) internet(1)
              security(5) mechanisms(5) pkix(7) id-mod(0)
              id-mod-pkixCommon-02(57) }

    id-pe
    FROM PKIX1Explicit-2009
            { iso(1) identified-organization(3) dod(6) internet(1)
              security(5) mechanisms(5) pkix(7) id-mod(0)
              id-mod-pkix1-explicit-02(51) } ;
    MUDCertExtensions EXTENSION ::= { ext-MUDURL, ... }
    ext-MUDURL EXTENSION ::= { SYNTAX MUDURLSyntax
    IDENTIFIED BY id-pe-mud-url }

    id-pe-mud-url OBJECT IDENTIFIER ::= { id-pe 25 }

    MUDURLSyntax ::= IA5String

    END

<CODE ENDS>
```

While this extension can appear in either an 802.AR manufacturer certificate (IDevID) or deployment certificate (LDevID), of course it is not guaranteed in either, nor is it guaranteed to be carried over. It is RECOMMENDED that MUD controller implementations maintain a table that maps a device to its MUD-URL.

## 11. The Manufacturer Usage Description LLDP extension

The IEEE802.1AB Link Layer Discovery Protocol (LLDP) is a one hop vendor-neutral link layer protocols used by end hosts network devices for advertising their identity, capabilities, and neighbors on an IEEE 802 local area network. Its Type-Length-Value (TLV) design allows for 'vendor-specific' extensions to be defined. IANA has a registered IEEE 802 organizationally unique identifier (OUI) defined as documented in [RFC7042]. The MUD LLDP extension uses a subtype defined in this document to carry the MUD URL.

The LLDP vendor specific frame has the following format:

```
+--------+--------+----------+---------+--------------
|TLV Type|  len   |   OUI    |subtype  | MUD URL
|  =127  |        |= 00 00 5E|  = 1    |
|(7 bits)|(9 bits)|(3 octets)|(1 octet)|(1-255 octets)
+--------+--------+----------+---------+--------------
```

where:

o  TLV Type = 127 indicates a vendor-specific TLV

o  len - indicates the TLV string length

o  OUI = 00 00 5E is the organizationally unique identifier of IANA

o  subtype = 1 (to be assigned by IANA for the MUD URL)

o  MUD URL - the length MUST NOT exceed 255 octets

The intent of this extension is to provide both a new device classifier to the network as well as some recommended configuration to the routers that implement policy. However, it is entirely the purview of the network system as managed by the network administrator to decide what to do with this information. The key function of this extension is simply to identify the type of device to the network in a structured way such that the policy can be easily found with existing toolsets.

Hosts, routers, or other network devices that implement this option are intended to have at most one MUD URL associated with them, so they may transmit at most one MUD URL value.

Hosts, routers, or other network devices that implement this option may ignore these options or take action based on receipt of these options. For example they may fill in information in the respective extensions of the LLDP Management Information Base (LLDP MIB). LLDP operates in a one-way direction. LLDPDUs are not exchanged as information requests by one device and response sent by another device. The other devices do not acknowledge LLDP information received from a device. No specific network behavior is guaranteed. When a device consumes this extension, it may either forward the URL and relevant remote device information to a MUD controller, or it will retrieve the usage description by resolving the URL.

## 12. Creating and Processing of Signed MUD Files

Because MUD files contain information that may be used to configure network access lists, they are sensitive. To insure that they have not been tampered with, it is important that they be signed. We make use of DER-encoded Cryptographic Message Syntax (CMS) [RFC5652] for this purpose.

### 12.1. Creating a MUD file signature

A MUD file MUST be signed using CMS as an opaque binary object. In order to make successful verification more likely, intermediate certificates SHOULD be included. The signature is stored at the same location as the MUD URL but with the suffix of ".p7s". Signatures are transferred using content-type "Application/pkcs7-signature".

For example:

```
% openssl cms -sign -signer mancertfile -inkey mankey \
              -in mudfile -binary -outform DER - \
              -certfile intermediatecert -out mudfile.p7s
```

Note: A MUD file may need to be re-signed if the signature expires.

### 12.2. Verifying a MUD file signature

Prior to retrieving a MUD file the MUD controller SHOULD retrieve the MUD signature file using the MUD URL with a suffix of ".p7s". For example, if the MUD URL is "https://example.com/.well-known/v1/modela", the MUD signature URL will be "https://example.com/.well-known/v1/modela.p7s".

Upon retrieving a MUD file, a MUD controller MUST validate the signature of the file before continuing with further processing. A MUD controller SHOULD produce an error and it MUST cease all processing of that file if the signature cannot be validated. It is important that MUD controllers have some reason to trust the certificates they are seeing. Therefore, it is RECOMMENDED that new signers be validated either directly by an administrator or by a service that has some reason to believe that the signer is a good actor.

For Example:

```
% openssl cms -verify -in mudfile.p7s -inform DER -content mudfile
```

Note the additional step of verifying the common trust root.

## 13. Extensibility

One of our design goals is to see that MUD files are able to be understood by as broad a cross-section of systems as is possible. Coupled with the fact that we have also chosen to leverage existing mechanisms, we are left with no ability to negotiate extensions and a limited desire for those extensions in any event. A such, a two-tier extensibility framework is employed, as follows:

1. At a coarse grain, a protocol version is included in a MUD URL. This memo specifies MUD version 1. Any and all changes are entertained when this version is bumped. Transition approaches between versions would be a matter for discussion in future versions.

2. At a finer grain, only extensions that would not incur additional risk to the device are permitted. Specifically, augmenting of the metainfo container is permitted with the understanding that such additions may be ignored. In addition, augmentation of the ACL model is permitted so long as it remains safe for a given ACE to be ignored by the MUD Controller or the network elements it configures. Most specifically, is is not permitted to include as an augmentation that modifies "deny" behavior without bumping the version. Furthermore, implementations that are not able to parse a component of the ACE array MUST ignore the entire array entry (e.g., not the entire array) and MAY ignore the entire MUD file.

## 14. Deployment Considerations

Because MUD consists of a number of architectural building blocks, it is possible to assemble different deployment scenarios. One key aspect is where to place policy enforcement. In order to protect the device from other devices within a local deployment, policy can be enforced on the nearest switch or access point. In order to limit unwanted traffic within a network, it may also be advisable to enforce policy as close to the Internet as possible. In some circumstances, policy enforcement may not be available at the closest hop. At that point, the risk of so-called east-west infection is increased to the number of devices that are able to communicate without protection.

A caution about some of the classes: admission of a device into the "manufacturer" and "same-manufacturer" class may have impact on access of other devices. Put another way, the admission may grow the access-list on switches connected to other devices, depending on how access is managed. Therefore, care should be given on managing that access-list growth. Alternative methods such as additional segmentation can be used to keep that growth within reason.

## 15. Security Considerations

Based on how a MUD-URL is emitted, a device may be able to lie about what it is, thus gaining additional network access. There are several means to limit risk in this case. The most obvious is to only believe devices that make use of certificate-based authentication such as IEEE 802.1AR certificates. When those certificates are not present, devices claiming to be of a certain manufacturer SHOULD NOT be included in that manufacturer grouping without additional validation of some form. This will occur when it makes use of primitives such as "manufacturer" for the purpose of accessing devices of a particular type. Similarly, network management systems may be able to fingerprint the device. In such cases, the MUD-URL can act as a classifier that can be proven or disproven. Fingerprinting may have other advantages as well: when 802.1AR certificates are used, because they themselves cannot change, fingerprinting offers the opportunity to add artificats to the MUD-URL. The meaning of such artifacts is left as future work.

Network management systems SHOULD NOT accept a usage description for a device with the same MAC address that has indicated a change of authority without some additional validation (such as review of the class). New devices that present some form of unauthenticated MUD URL SHOULD be validated by some external means when they would be otherwise be given increased network access.

It may be possible for a rogue manufacturer to inappropriately exercise the MUD file parser, in order to exploit a vulnerability. There are three recommended approaches to address this threat. The first is to validate the signature of the MUD file. The second is to have a system do a primary scan of the file to ensure that it is both parseable and believable at some level. MUD files will likely be relatively small, to start with. The number of ACEs used by any given device should be relatively small as well. It may also be useful to limit retrieval of MUD URLs to only those sites that are known to have decent web reputations.

Use of a URL necessitates the use of domain names. If a domain name changes ownership, the new owner of that domain may be able to provide MUD files that MUD controllers would consider valid. There are a few approaches that can mitigate this attack. First, MUD controllers SHOULD cache certificates used by the MUD file server. When a new certificate is retrieved for whatever reason, the MUD controller should check to see if ownership of the domain has changed. A fair programmatic approximation of this is when the name servers for the domain have changed. If the actual MUD file has changed, the controller MAY check the WHOIS database to see if registration ownership of a domain has changed. If a change has occured, or if for some reason it is not possible to determine whether ownership has changed, further review may be warranted. Note, this remediation does not take into account the case of a device that was produced long ago and only recently fielded, or the case where a new MUD controller has been installed.

The release of a MUD URL by a device reveals what the device is, and provides an attacker with guidance on what vulnerabilities may be present.

While the MUD URL itself is not intended to be unique to a specific device, the release of the URL may aid an observer in identifying individuals when combined with other information.  This is a privacy consideration.

In addressing both of these concerns, implementors should take into account what other information they are advertising through mechanisms such as mDNS[RFC6872], how a device might otherwise be identified, perhaps through how it behaves when it is connected to the network, whether a device is intended to be used by individuals or carry personal identifying information, and then apply appropriate data minimization techniques.  One approach is to make use of TEAP [RFC7170] as the means to share information with authorized components in the network.  Network devices may also assist in limiting access to the MUD-URL through the use of mechanisms such as DHCPv6-Shield [RFC7610].

Please note that the security considerations mentioned in Section 4.7 of [I-D.ietf-netmod-rfc6087bis] are not applicable in this case because the YANG serialization is not intended to be accessed via NETCONF.  However, for those who try to instantiate this model in a device via NETCONF, all objects in each model in this draft exhibit similar security characteristics as [I-D.ietf-netmod-acl-model].  The basic purpose of MUD is to configure access, and so by its very nature can be disruptive if used by unauthorized parties.

# 16.  IANA Considerations

## 16.1.  YANG Module Registrations

The following YANG modules are requested to be registred in the "IANA Module Names" registry:

The ietf-mud module:

- o  Name: ietf-mud
- o  XML Namespace: urn:ietf:params:xml:ns:yang:ietf-mud

o  Prefix: ief-mud

o  Reference: This memo

The ietf-acldns module:

o  Name: ietf-acldns

o  XML Namespace: urn:ietf:params:xml:ns:yang:ietf-acldns

o  Prefix: ietf-acldns

o  Reference: This memo

## 16.2.  DHCPv4 and DHCPv6 Options

The IANA has allocated option 161 in the Dynamic Host Configuration Protocol (DHCP) and Bootstrap Protocol (BOOTP) Parameters registry for the MUD DHCPv4 option.

IANA is requested to allocated the DHCPv4 and v6 options as specified in Section 9.

## 16.3.  PKIX Extensions

IANA is kindly requested to make the following assignments for:

o The MUDURLExtnModule-2016 ASN.1 module in the "SMI Security for PKIX Module Identifier" registry (1.3.6.1.5.5.7.0).

o id-pe-mud-url object identifier from the "SMI Security for PKIX Certificate Extension" registry (1.3.6.1.5.5.7.1).

The use fo these values is specified in Section 10.

## 16.4.  Well Known URI Suffix

The IANA has allocated the URL suffix of "mud" as follows:

o URI Suffix: "mud" o Specification documents: this document o Related information: n/a

## 16.5.  MIME Media-type Registration for MUD files

The following media-type is defined for transfer of MUD file:

- o Type name: application
- o Subtype name: mud+json
- o Required parameters: n/a
- o Optional parameters: n/a
- o Encoding considerations: 8bit; application/mud+json values are represented as a JSON object; UTF-8 encoding SHOULD be employed.
- o Security considerations: See {{secon}} of this document.
- o Interoperability considerations: n/a
- o Published specification: this document
- o Applications that use this media type: MUD controllers as specified by this document.
- o Fragment identifier considerations: n/a
- o Additional information:

  Magic number(s): n/a
  File extension(s): n/a
  Macintosh file type code(s): n/a

- o Person & email address to contact for further information: Eliot Lear <lear@cisco.com>, Ralph Droms <rdroms@cisco.com>
- o Intended usage: COMMON
- o Restrictions on usage: none
- o Author:
   Eliot Lear <lear@cisco.com>
   Ralph Droms <rdroms@cisco.com>
- o Change controller: IESG
- o Provisional registration? (standards tree only): No.

## 16.6.  LLDP IANA TLV Subtype Registry

IANA is requested to create a new registry for IANA Link Layer Discovery Protocol (LLDP) TLV subtype values.  The recommended policy for this registry is Expert Review.  The maximum number of entries in the registry is 256.

IANA is required to populate the initial registry with the value:

LLDP subtype value = 1 (All the other 255 values should be initially marked as 'Unassigned'.)

Description = the Manufacturer Usage Description (MUD) Uniform Resource Locator (URL)

Reference = < this document >

### 16.7. The MUD Well Known Universal Resource Name (URNs)

The following parameter registry is requested to be added in accordance with [RFC3553]

```
Registry name: "urn:ietf:params:mud" is requested.
Specification: this document
Repository: this document
Index value:  Encoded identically to a TCP/UDP port service
              name, as specified in Section 5.1 of [RFC6335]
```

The following entries should be added to the "urn:ietf:params:mud" name space:

"urn:ietf:params:mud:dns" refers to the service specified by [RFC1123]. "urn:ietf:params:mud:ntp" refers to the service specified by [RFC5905].

## 17. Acknowledgments

The authors would like to thank Einar Nilsen-Nygaard, Bernie Volz, Tom Gindin, Brian Weis, Sandeep Kumar, Thorsten Dahm, John Bashinski, Steve Rich, Jim Bieda, and Dan Wing for their valuable advice and reviews. Russ Housley entirely rewrote Section 10 to be a complete module. Adrian Farrel provided the basis for privacy considerations text. Kent Watson provided a thorough review of the archictecture and the YANG model. The remaining errors in this work are entirely the responsibility of the author.

## 18. References

### 18.1. Normative References

[I-D.ietf-anima-bootstrapping-keyinfra]
Pritikin, M., Richardson, M., Behringer, M., Bjarnason, S., and K. Watsen, "Bootstrapping Remote Secure Key Infrastructures (BRSKI)", draft-ietf-anima-bootstrapping-keyinfra-05 (work in progress), March 2017.

[I-D.ietf-netmod-acl-model]
Bogdanovic, D., Koushik, K., Huang, L., and D. Blair, "Network Access Control List (ACL) YANG Data Model", draft-ietf-netmod-acl-model-10 (work in progress), March 2017.

[I-D.ietf-netmod-rfc6087bis]
Bierman, A., "Guidelines for Authors and Reviewers of YANG Data Model Documents", draft-ietf-netmod-rfc6087bis-12 (work in progress), March 2017.

[IEEE8021AB]
Institute for Electrical and Electronics Engineers, "IEEE Standard for Local and Metropolitan Area Networks-- Station and Media Access Control Connectivity Discovery", n.d..

[RFC1123] Braden, R., Ed., "Requirements for Internet Hosts - Application and Support", STD 3, RFC 1123, DOI 10.17487/RFC1123, October 1989, <http://www.rfc-editor.org/info/rfc1123>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, DOI 10.17487/RFC2131, March 1997, <http://www.rfc-editor.org/info/rfc2131>.

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, DOI 10.17487/RFC2818, May 2000, <http://www.rfc-editor.org/info/rfc2818>.

[RFC3315] Droms, R., Ed., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, DOI 10.17487/RFC3315, July 2003, <http://www.rfc-editor.org/info/rfc3315>.

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, Ed., "Extensible Authentication Protocol (EAP)", RFC 3748, DOI 10.17487/RFC3748, June 2004, <http://www.rfc-editor.org/info/rfc3748>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <http://www.rfc-editor.org/info/rfc3986>.

[RFC3987] Duerst, M. and M. Suignard, "Internationalized Resource Identifiers (IRIs)", RFC 3987, DOI 10.17487/RFC3987, January 2005, <http://www.rfc-editor.org/info/rfc3987>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <http://www.rfc-editor.org/info/rfc5280>.

[RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, DOI 10.17487/RFC5652, September 2009, <http://www.rfc-editor.org/info/rfc5652>.

[RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <http://www.rfc-editor.org/info/rfc5905>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <http://www.rfc-editor.org/info/rfc6020>.

[RFC6092] Woodyatt, J., Ed., "Recommended Simple Security Capabilities in Customer Premises Equipment (CPE) for Providing Residential IPv6 Internet Service", RFC 6092, DOI 10.17487/RFC6092, January 2011, <http://www.rfc-editor.org/info/rfc6092>.

[RFC6335] Cotton, M., Eggert, L., Touch, J., Westerlund, M., and S. Cheshire, "Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry", BCP 165, RFC 6335, DOI 10.17487/RFC6335, August 2011, <http://www.rfc-editor.org/info/rfc6335>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <http://www.rfc-editor.org/info/rfc6991>.

[RFC7227] Hankins, D., Mrugalski, T., Siodelski, M., Jiang, S., and S. Krishnan, "Guidelines for Creating New DHCPv6 Options", BCP 187, RFC 7227, DOI 10.17487/RFC7227, May 2014, <http://www.rfc-editor.org/info/rfc7227>.

[RFC7610] Gont, F., Liu, W., and G. Van de Velde, "DHCPv6-Shield: Protecting against Rogue DHCPv6 Servers", BCP 199, RFC 7610, DOI 10.17487/RFC7610, August 2015, <http://www.rfc-editor.org/info/rfc7610>.

## 18.2. Informative References

[FW95] Chapman, D. and E. Zwicky, "Building Internet Firewalls", January 1995.

[IEEE8021AR] Institute for Electrical and Electronics Engineers, "Secure Device Identity", 1998.

[ISO.8601.1988] International Organization for Standardization, "Data elements and interchange formats - Information interchange - Representation of dates and times", ISO Standard 8601, June 1988.

[RFC1984] IAB and IESG, "IAB and IESG Statement on Cryptographic Technology and the Internet", BCP 200, RFC 1984, DOI 10.17487/RFC1984, August 1996, <http://www.rfc-editor.org/info/rfc1984>.

[RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, DOI 10.17487/RFC3339, July 2002, <http://www.rfc-editor.org/info/rfc3339>.

[RFC3553] Mealling, M., Masinter, L., Hardie, T., and G. Klyne, "An IETF URN Sub-namespace for Registered Protocol Parameters", BCP 73, RFC 3553, DOI 10.17487/RFC3553, June 2003, <http://www.rfc-editor.org/info/rfc3553>.

[RFC6872] Gurbani, V., Ed., Burger, E., Ed., Anjali, T., Abdelnur, H., and O. Festor, "The Common Log Format (CLF) for the Session Initiation Protocol (SIP): Framework and Information Model", RFC 6872, DOI 10.17487/RFC6872, February 2013, <http://www.rfc-editor.org/info/rfc6872>.

[RFC7042] Eastlake 3rd, D. and J. Abley, "IANA Considerations and IETF Protocol and Documentation Usage for IEEE 802 Parameters", BCP 141, RFC 7042, DOI 10.17487/RFC7042, October 2013, <http://www.rfc-editor.org/info/rfc7042>.

[RFC7170] Zhou, H., Cam-Winget, N., Salowey, J., and S. Hanna, "Tunnel Extensible Authentication Protocol (TEAP) Version 1", RFC 7170, DOI 10.17487/RFC7170, May 2014, <http://www.rfc-editor.org/info/rfc7170>.

[RFC7452] Tschofenig, H., Arkko, J., Thaler, D., and D. McPherson, "Architectural Considerations in Smart Object Networking", RFC 7452, DOI 10.17487/RFC7452, March 2015, <http://www.rfc-editor.org/info/rfc7452>.

[RFC7488] Boucadair, M., Penno, R., Wing, D., Patil, P., and T. Reddy, "Port Control Protocol (PCP) Server Selection", RFC 7488, DOI 10.17487/RFC7488, March 2015, <http://www.rfc-editor.org/info/rfc7488>.

## Appendix A.  Changes from Earlier Versions

RFC Editor to remove this section prior to publication.

Draft -05 to -06:

- Make clear that this is a component architecture (Polk and Watson)
- Add order of operations (Watson)
- Add extensions leaf-list (Pritikin)
- Remove previous-mud-file (Watson)
- Modify text in last-update (Watson)
- Clarify local networks (Weis, Watson)
- Fix contact info (Watson)
- Terminology clarification (Weis)
- Advice on how to handle LDevIDs (Watson)
- Add deployment considerations (Watson)
- Add some additional text about fingerprinting (Watson)
- Appropriate references to 6087bis (Watson)
- Change systeminfo to a URL to be referenced (Lear)

Draft -04 to -05: * syntax error correction

Draft -03 to -04: * Re-add my-controller

Draft -02 to -03: * Additional IANA updates * Format correction in YANG.  * Add reference to TEAP.

Draft -01 to -02: * Update IANA considerations * Accept Russ Housley rewrite of X.509 text * Include privacy considerations text * Redo the URL limit.  Still 255 bytes, but now stated in the URL definition.  * Change URI registration to be under urn:ietf:params

Draft -00 to -01: * Fix cert trust text.  * change supportInformation to meta-info * Add an informaitonal element in.  * add urn registry and create first entry * add default elements

## Appendix B.  Default MUD elements

What follows is a MUD file that permits DNS traffic to a controller that is registered with the URN "urn:ietf:params:mud:dns" and traffic NTP to a controller that is registered "urn:ietf:params:mud:ntp". This is considered the default behavior and the ACEs are in effect appended to whatever other ACEs.  To block DNS or NTP one repeats the matching statement but replace "permit" with deny.  Because ACEs are processed in the order they are received, the defaults would not be reached.  A MUD controller might further decide to optimize to simply not include the defaults when they are overriden.

A complete MUD entry is included below.


```
{
  "ietf-mud:metainfo": {
    "last-update": "2016-09-27T15:10:24+02:00",
    "cache-validity": 168
  },
  "acl:access-lists": {
    "access-list": [
      {
        "acl-name": "mud-53134-v4in",
        "acl-type": "ipv4-acl",
        "ietf-mud:packet-direction": "to-device",
        "access-list-entries": {
          "ace": [
            {
              "rule-name": "entout0-in",
              "matches": {
                "ietf-mud:controller": "urn:ietf:params:mud:dns",
                "protocol": 17,
                "source-port-range": {
                  "lower-port": 53,
                  "upper-port": 53
                }
              },
              "actions": {
                "permit": [
```

```
                    null
                  ]
                }
              },
              {
                "rule-name": "entout1-in",
                "matches": {
                  "ietf-mud:controller": "urn:ietf:params:mud:dns",
                  "protocol": 6,
                  "source-port-range": {
                    "lower-port": 53,
                    "upper-port": 53
                  }
                },
                "actions": {
                  "permit": [
                    null
                  ]
                }
              },
              {
                "rule-name": "entout2-in",
                "matches": {
                  "ietf-mud:controller": "urn:ietf:params:mud:ntp",
                  "protocol": 17,
                  "source-port-range": {
                    "lower-port": 123,
                    "upper-port": 123
                  }
                },
                "actions": {
                  "permit": [
                    null
                  ]
                }
              }
            ]
          }
        },
        {
          "acl-name": "mud-53134-v4out",
          "acl-type": "ipv4-acl",
          "ietf-mud:packet-direction": "from-device",
          "access-list-entries": {
            "ace": [
              {
                "rule-name": "entout0-in",
                "matches": {
```

```
                  "ietf-mud:controller": "urn:ietf:params:mud:dns",
                  "protocol": 17,
                  "source-port-range": {
                    "lower-port": 53,
                    "upper-port": 53
                  }
                },
                "actions": {
                  "permit": [
                    null
                  ]
                }
              },
              {
                "rule-name": "entout1-in",
                "matches": {
                  "ietf-mud:controller": "urn:ietf:params:mud:dns",
                  "protocol": 6,
                  "source-port-range": {
                    "lower-port": 53,
                    "upper-port": 53
                  }
                },
                "actions": {
                  "permit": [
                    null
                  ]
                }
              },
              {
                "rule-name": "entout2-in",
                "matches": {
                  "ietf-mud:controller": "urn:ietf:params:mud:ntp",
                  "protocol": 17,
                  "source-port-range": {
                    "lower-port": 123,
                    "upper-port": 123
                  }
                },
                "actions": {
                  "permit": [
                    null
                  ]
                }
              }
            ]
          }
        },
```

```
         {
           "acl-name": "mud-53134-v6in",
           "acl-type": "ipv6-acl",
           "ietf-mud:packet-direction": "to-device",
           "access-list-entries": {
             "ace": [
               {
                 "rule-name": "entout0-in",
                 "matches": {
                   "ietf-mud:controller": "urn:ietf:params:mud:dns",
                   "protocol": 17,
                   "source-port-range": {
                     "lower-port": 53,
                     "upper-port": 53
                   }
                 },
                 "actions": {
                   "permit": [
                     null
                   ]
                 }
               },
               {
                 "rule-name": "entout1-in",
                 "matches": {
                   "ietf-mud:controller": "urn:params:mud:dns",
                   "protocol": 6,
                   "source-port-range": {
                     "lower-port": 53,
                     "upper-port": 53
                   }
                 },
                 "actions": {
                   "permit": [
                     null
                   ]
                 }
               },
               {
                 "rule-name": "entout2-in",
                 "matches": {
                   "ietf-mud:controller": "urn:ietf:params:mud:ntp",
                   "protocol": 17,
                   "source-port-range": {
                     "lower-port": 123,
                     "upper-port": 123
                   }
                 },
```

```
                 "actions": {
                   "permit": [
                     null
                   ]
                 }
               }
             ]
           }
         },
         {
           "acl-name": "mud-53134-v6out",
           "acl-type": "ipv6-acl",
           "ietf-mud:packet-direction": "from-device",
           "access-list-entries": {
             "ace": [
               {
                 "rule-name": "entout0-in",
                 "matches": {
                   "ietf-mud:controller": "urn:ietf:params:mud:dns",
                   "protocol": 17,
                   "source-port-range": {
                     "lower-port": 53,
                     "upper-port": 53
                   }
                 },
                 "actions": {
                   "permit": [
                     null
                   ]
                 }
               },
               {
                 "rule-name": "entout1-in",
                 "matches": {
                   "ietf-mud:controller": "urn:ietf:params:mud:dns",
                   "protocol": 6,
                   "source-port-range": {
                     "lower-port": 53,
                     "upper-port": 53
                   }
                 },
                 "actions": {
                   "permit": [
                     null
                   ]
                 }
               },
               {
```

```
                 "rule-name": "entout2-in",
                 "matches": {
                   "ietf-mud:controller": "urn:ietf:params:mud:ntp",
                   "protocol": 17,
                   "source-port-range": {
                     "lower-port": 123,
                     "upper-port": 123
                   }
                 },
                 "actions": {
                   "permit": [
                     null
                   ]
                 }
               }
             ]
           }
         }
       ]
     }
   }
```


## Authors' Addresses


Eliot Lear
Cisco Systems
Richtistrasse 7
Wallisellen  CH-8304
Switzerland

Phone: +41 44 878 9200
Email: lear@cisco.com

Ralph Droms

Phone: +1 978 376 3731
Email: rdroms@gmail.com

Dan Romascanu

Phone: +972 54 5555347
Email: dromasca@gmail.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: September 14, 2017

Z. Li
Huawei Technologies
Y. Zheng
China Unicom
J. Zhang
S. Xu
Huawei Technologies
March 13, 2017

# An Architecture of Network Artificial Intelligence(NAI)
draft-li-opsawg-network-ai-arch-00

## Abstract

Artificial intelligence is an important technical trend in the industry. With the development of network, it is necessary to introduce artificial intelligence technology to achieve self-adjustment, self- optimization, self-recovery of the network through collection of huge data of network state and machine learning. This draft defines the architecture of Network Artificial Intelligence (NAI), including the key components and the key protocol extension requirements.

## Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described inRFC 2119 [RFC2119]

Table of Contents



## 1. Introduction

Artificial Intelligence is an important technical trend in the industry. The two key aspects of Artificial Intelligence are perception and cognition. Artificial Intelligence has evolved from an early non-learning expert system to a learning-capable machine learning era. In recent years, the rapid development of the deep learning branch based on the neural network and the maturity of the big data technology and software distributed architecture make the Artificial Intelligence in many fields (such as transportation, medical treatment, education, etc.) have been applied. With the development of network, it is necessary to introduce artificial intelligence technology to achieve self-adjustment, self-optimization, self-recovery of the network through collection of huge data of network state and machine learning. The areas of machine learning which are easier to be used in the network field may include: root cause analysis of network failures, network traffic prediction, traffic adjustment and optimization, security defense, security auditing, etc., to implement network perception and cognition.

This draft defines the architecture of Network Artificial Intelligence (NAI), including the key components and the key protocol extension requirements.

## 2. Terminology

AI: Artificial Intelligence

NAI: Network Artificial Intelligence

## 3. Architecture

```
                ^                                   ^
             (4)|                                   |(4)
+---------------|--------------+    +---------------|--------------+
| Domain 1      |              |    |               |     Domain 2 |
|        +------------+        |    |        +------------+        |
|        |  Central   |        |    |        |  Central   |        |
|     (1)| Controller |----------------------| Controller |(1)     |
|        |    with    |        |    |        |    with    |        |
|        |     NTA    |        |    |        |     NTA    |        |
|        +------------+        |    |        +------------+        |
|         /          \         |    |         /          \         |
|     (3)/            \        |    |        /            \(3)     |
|       /              \       |    |       /              \       |
| +--------+        +--------+ |    | +--------+        +--------+ |
| |        |        |        | |    | |        |        |        | |
| |Network | ...... |Network | |    | |Network | ...... |Network | |
| | Device |  (2)   | Device | |    | | Device |  (2)   | Device | |
| |    1   |        |    N   | |    | |    1   |        |    N   | |
| +--------+        +--------+ |    | +--------+        +--------+ |
|                              |    |                              |
+------------------------------+    +------------------------------+
```

Figure 1: An Architecture of Network Artificial Intelligence(NAI)

The architecture of Network artificial intelligence includes following key components:

(1) Central Controller: Centralized controller is the core part of Network Artificial Intelligence which can be called as 'Network

Brain'. The Network Telemetry and Analytics (NTA) engines can be introduced acompanying with the central controller. The Network Telemetry and Analytics (NTA) engine inclues data collector, analytics framework, data persistence, and NAI applications.

(2) Network Device: IP network operation and maintenance are always a big challenge since the network can only provide limited state information. The network states includes but are not limited to topology, traffic engineering, operation and maintenance information, network failure information and related information to locate the network failure. In order to provide these information, the network must be able to support more OAM mechanisms to acquire more state information and report to the controller. Then the controller can get the complete state information of the network which is the base of Network Artificial Intelligence(NAI).

(3) Southbound Protocol and Models of Controller: As network devices provide huge network state information, it proposes a number of new requirements for protocols and models between controllers and network devices. The traditional southbound protocol such as Netconf and SNMP can not meet the performance requirements. It is necessary to introduce some new high-performance protocols to collect network state data. At the same time, the models of network data should be completed. Moreover with the introduction of new OAM mechanisms of network devices, new models of network data should be introduced.

(4) Northbound Model of Controller: The goal of the Network Artificial Intelligence is to reduce the technical requirements on the network administrators and release them from the heavy network management, control, maintenance work. The abstract northbound model of the controller for different network services should be simple and easy to be understood.

## 4. Process

NAI consists of following processes:

-- Data Collection

From the time aspect, data collection can be divided into real-time data collection and non-real-time collection.

From the content aspect, data collection can be divided into network information collection (including topology, tunnels, routing, equipment configuration, etc.) and traffic collection (the collection network traffic, network load, device KPI, etc.).

-- Data Storage

Store data collected from network. Many existing big data storage technologies can be used here.

-- Data Processing

This is preliminary data processing too select effective data and simply analyse data relationship.

-- Analyse

Analyse engine will provide the data analysis results using machine learning algorithm.

-- Closed Loop Control

According to the results of intelligent analysis and policy set by user, the central controller will implement closed-loop control of the network.

## 5. Classification

NAI can be divided into off-line process and on-line process in accordance to the time aspect of the data collection and analysis.

Off-line process refers to process of the existing data, or non-real-time collection data. Although the analysis process will also focus on the relationship between data and time, but it does not require real-time analysis. Off-line process is mainly used for two purposes: (1) training or verification of real-time process design; (2) trouble shooting or reason analysis for events that have already occurred.

On-line process is efficient real-time collection, processing and analysis of the data, to operate network monitoring and event forecasting. The main purpose of the on-line process are: (1) network capacity monitoring and precise optimizing; (2) network event prediction and fast trouble shooting; (3) real-time network optimization according to the policy.

## 6. Requirement of Protocol Extensions

### 6.1. Requirement of Southbound Protocols

REQ 01: The southbound protocol of the controller should be introduced to meet the performance requirements of collecting huge data of network states.

The soundbound protocol can be based on the extensions of the existing traditional protocols such link state colloction protocols, PCEP[RFC5440], BMP[RFC7854], etc. Or the new protocol like Telemetry[I-D.kumar-rtgwg-grpc-protocol] can be introduced as the soundbound protocols. The protocol choice will be based on the application scenarios of NAI.

### 6.2. Requirement of Data Collection

REQ 02: The data collected from the network devices includes but not limites to following information:

-- network topology information

-- routing protocol status

-- IP routes and MAC routes

-- LSP information

-- network traffic inforamtion

-- network configuration

-- network device KPIs

-- log of network elements

-- trap of network elements

-- OAM information

### 6.3. Requirement of Devices

REQ 03: New OAM mechanisms should be introduced for the network devices in order to acquire more types of network state data.

### 6.4. Requirement of Northbound Interface

REQ 04: The abstract network-based service models should be provided by the controller as the northbound models to satisfy the requirements of different services.

## 7. IANA Considerations

This document makes no request of IANA.

## 8. Security Considerations

TBD.

## 9. Normative References

[I-D.kumar-rtgwg-grpc-protocol]
Kumar, A., Kolhe, J., Ghemawat, S., and L. Ryan, "gRPC Protocol", draft-kumar-rtgwg-grpc-protocol-00 (work in progress), July 2016.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC5440] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, DOI 10.17487/RFC5440, March 2009, <http://www.rfc-editor.org/info/rfc5440>.

[RFC7854] Scudder, J., Ed., Fernando, R., and S. Stuart, "BGP Monitoring Protocol (BMP)", RFC 7854, DOI 10.17487/RFC7854, June 2016, <http://www.rfc-editor.org/info/rfc7854>.

## Authors' Addresses

Zhenbin Li
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: lizhenbin@huawei.com

Yi Zheng
China Unicom
No.9, Shouti Nanlu, Haidian District
Beijing 100048
China

Email: zhengyi39@chinaunicom.cn

Jinhui Zhang
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: jason.zhangjinhui@huawei.com

Xu Shiping
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
P.R. China

Email: xushiping7@huawei.com

Operations and Management Area Working Group
Internet-Draft
Intended status: Informational
Expires: August 28, 2017

B. Pularikkal
Cisco Systems
T. Pauly
Apple Inc.
M. Grayson
S. Gundavelli
Cisco Systems
S. Touati
Ericsson

February 24, 2017

# Carrier Wi-Fi Calling Deployment Considerations
draft-pularikkal-opsawg-wifi-calling-02

## Abstract

Carrier Wi-Fi Calling is a solution that allows mobile operators to seamlessly offload mobile voice signaling and bearer traffic onto Wi-Fi access networks, which may or may not be managed by the mobile operators. Mobile data offload onto Wi-Fi access networks has already become very common, as Wi-Fi access has become more ubiquitous. However, the offload of mobile voice traffic onto Wi-Fi networks has become prevalent only in recent years. This was primarily driven by the native Wi-Fi Calling client support introduced by device vendors. The objective of this document is to provide a high level deployment reference to Mobile Operators and Wi-Fi Operators on Carrier Wi-Fi Calling.

## Status of This Memo

Copyright Notice



Table of Contents

# 1. Introduction

There are several SP Managed and Over the Top Voice Solutions deployed today which can leverage Wi-Fi access networks. Some of these solutions rely on standalone applications installed on the Mobile Handset and other Mobile devices such as tablets. Also there are solutions, which leverage dedicated hardware built exclusively to support Voice over Wi-Fi.e.g,in enterprise type environments. The scope of this document is VoWiFi solutions, which are deployed by Mobile Network Operators also known as Wireless Carriers. VoWiFi from the context of Mobile Voice offload is often referred to as Carrier Wi-Fi Calling. The deployment of Carrier Wi-Fi Calling requires some kind of integration between the Wi-Fi Access network and Mobile Packet Core. Carrier Wi-Fi calling solutions deployed today predominantly uses an 'untrusted Wi-Fi'model that delivers simple IP connectivity to facilitate Mobile Packet Core integration. With this 'untrusted' approach, Mobile Operators are able to make use of the existing Wi-Fi deployment footprint regardless of whether it is owned by the MNOs or by their roaming partners or Wi-Fi Operators without any kind of partnership with the MNOs. This model has definitely allowed MNOs to accelerate the adoption of Wi-Fi calling. However, this comes with some caveats, as depending on the Wi-Fi network, there may be no visibility or control over it by the MNO, impacting its ability to carry voice calls without compromising end user experience.

It is in the interest of both MNOs as well as Wi-Fi Operators to improve the quality of experience for Wi-Fi Calling delivered over a Wi-Fi access network. MNOs have the incentive to make sure that the end user experience does not get compromised while the voice service is offloaded over Wi-Fi access. Wi-Fi operators have the business incentive to enter into roaming partnerships with the MNOs and support Wi-Fi calling with certain Service Level Agreements. In some deployments, it is possible for the MNOs to own some Wi-Fi hotspot deployments. In such cases, MNO will effectively be the Wi-Fi operator as well.

Objective of this document is to provide a Carrier Wi-Fi Calling deployment reference to Wi-Fi Operators and MNOs with primary focus on the Wi-Fi Access Network and the Wi-Fi to Packet Core integration aspects.

## 2. Terminology

Service Provider (SP)

Refers to a provider of telecommunications services such as Broadband Operator or Mobile Operator. An SP may provide several telecommunications services.

APP

Refers to computer program typically designed to run on Mobile devices such as smartphones and tablets.

Wireless Fidelity (Wi-Fi)

Technology that allows devices to wirelessly connect using 2.4 GHz and 5.0 GHz unlicensed radio bands. Wi-Fi is defined as part of IEEE 802.11 standards

Voice over Wi-Fi (VoWiFi)

Any solution, which supports voice services over Wi-Fi.

Mobile Network Operator (MNO)

A wireless communications service provider who owns and operates licensed wireless access network and the backend infrastructure to offer mobile voice, data and multimedia services.

3rd Generation Partnership Project (3GPP)

3GPP unites seven telecommunications standards development organizations known as Organizational Partners and provides their members with a stable environment to produce the reports and specifications that define 3GPP technologies

Groups Special Mobile Association (GSMA)

GSMA represents the interests of mobile operators worldwide, uniting nearly 800 operators with more than 250 companies in the broader mobile ecosystem, including handset and device makers, software companies, equipment providers and internet companies, as well as organizations in adjacent industry sectors.

User Equipment (UE)

Term represents any device used directly by an end user to communicate.

Wireless Local Area Network (WLAN)

Refers to IEEE 802.11 based Wi-Fi access networks and represents an extended service set consisting of multiple access points.

Long Term Evolution (LTE)

Is the fourth generation 3GPP standard set for wireless communication of mobile devices in end-to-end IP environment.

Evolved Packet Core (EPC)

Represents the Core Network in the 3GPP LTE system Architecture.

Packet Data Network (PDN)

PDN represents a network in the packet core a Mobile UE device wants to communicate with.  PDN generally is mapped to a set of related services.

Access Point Name (APN)

APN represents a set of services available to a specific PDN. Typically UE devices will be configured to access multiple APNs corresponding various services in the packet core.

Trusted WLAN Access Gateway (TWAG)

Performs the gateway function between a trusted WLAN access network and packet core.  It acts as the default gateway and DHCP Server for UE devices connected to the WLAN access network for trusted Wi-Fi to packet core integration model.

Evolved Packet Data Gateway (ePDG)

ePDG performs the gateway function between WLAN access network and Mobile Packet core in an untrusted model.  Main function of ePDG is to secure the data transmission with a UE connected to the EPC.

PDN Gateway (P-GW)

P-GW is the subscriber session anchor in EPC.  It enforces policy and also has a role in IP persistence in roaming scenarios.  Based up on the policy, P-GW steers traffic towards various PDN networks corresponding to various APNs.

IP Multi-Media Subsystem (IMS)

An Architectural framework for delivering IP multimedia services. And is defined in 3GPP

Policy and Charging Rule Function (PCRF)

A system in EPC, which detects service data flows, applies policies and QoS to subscriber flows to and supports flow based charging

Session Initiation Protocol (SIP)

SIP is an application layer control protocol that can establish, modify and terminate multimedia sessions or calls.

Real-time Transport Protocol (RTP)

RTP is a transport protocol, which provides end-to-end delivery services for data with real-time characteristics such as interactive audio and video.

Proxy Mobile IPv6 (PMIPv6)

PMIPv6 is a network based mobility management protocol standardized by IETF and adopted in 3GPP.

GPRS Tunneling Protocol (GTP)

Group of IP based communications protocols used in 3GPP architectures.

S2a Interface

Is the interface between TWAG and P-GW and can be either GTP or PMIPv6 based

S2b Interface

Interface between ePDG and P-GW and can be either GTP or PMIPv6

## 3. Architecture Overview

This section provides a very high level overview of the end-to-end Architecture for Carrier Wi-Fi Calling. It is outside the scope of this document to provide a detailed Architecture description, as all the functional entities and the protocol interfaces are well defined in the 3GPP and GSMA specifications [3GPPTS23.402,GSMAIR61,GSMAIR51]. Figure-01 below is used to describe the Architecture components at a high level.

```
                                        +-----+     +-----+
                                        | 3GPP+-----+ HSS |
                                        | AAA |     +-----+
                                        +-----+
                                           |
                                           |        +-----+
                                           |        | PCRF|
                                           |        +-----+
                                           |           |
                                       +--------+      |
                  +---+   +-----+  Backhaul| TWAG /|    +-----+
                  |UE +---+WLAN +----------+ ePDG  +----+ PGW |
                  +---+   +-----+         |       |    +-----+
                                       +--------+      |
                                                       |
                                                    +-----+
                                                    | IMS |
                                                    +-----+
```

Figure 1: High Level Architecture

The UE is the end user device such as a smartphone running native Wi-Fi Calling client. The UE is connected to a Wi-Fi access network, which is represented by the block WLAN in the diagram. Depending up on the trust model, TWAG or ePDG gateway is used to integrate the WLAN access network to the MNO packet core.More details around this untrusted and trusted approaches are covered in the next section. The P-GW acts as the common anchor for the subscriber sessions regardless of whether the UE is connected to Wi-Fi or LTE (not shown), allowing the preservation of the IP Session during a handover between LTE and Wi-Fi. IMS provides several functions related to SIP based call control signaling, namely SIP authentication, basic telephony services, supplementary services, interworking with other IMS systems, and offload into circuit switched voice networks. In addition to voice, the same IMS infrastructure may be leveraged for other multi-media functions such as video calling. The IMS framework consists of several functional entities and is omitted for the sake of simplicity here. PCRF performs classical Policy and Charging Rule functions in the Mobile Packet Core. For the Wi-Fi calling solution, it will trigger the establishment of the default and dedicated bearers on the S2a or S2b interfaces for SIP and RTP traffic between the PGW and the TWAG/ePDG.

# 4. Wi-Fi Calling Deployment Considerations

This section covers deployment considerations for an end-to-end Wi-Fi calling Architecture that can influence the quality of experience, availability and monetization aspects of the solution offering.

## 4.1. Wi-Fi to Packet Core Integration

There are three different Architecture options available for Wi-Fi to Packet Core integration for the deployment of Wi-Fi calling. Each of these models are described in the sub-sections below:

### 4.1.1. Untrusted Model

This model is built around the assumption that the Wi-Fi access network is ’unmanaged’ or untrusted from the MNOs perspective. Since this model does not rely on any security or data privacy implementations on the Wi-Fi access network, it requires the establishment of an IPSec tunnel between the UE device and the Mobile Packet Core. The ePDG gateway acts as the IPSec tunnel termination point on the packet core side. The ePDG handles the user authentication as well as the establishment of an S2b packet data network connection towards the P-GW using the GTP based S2b interface. This Architecture model is illustrated in figure-2 below.

```
      +--------------+            +----------+          +-------+
      | +----------+ |  IMS-APN   |          |          |       |
      | |    SWu   | |  Traffic   |          |          |       |
      | |  Client  +------------------------------------+       |
      | |          | | Other APN  |          |          | ePDG  |
      | |          | |  traffic   |          |          |       |
      | |          +------------------------------------+       |
      | |          | |            |          |          +-------+
      | +----------+ |            |          |             |  |
      |              |            |          |         S2b|  |S2b
      |     UE       |            |  WLAN    |             |  +---+
      |              |            | Access   |             |      |
      | +----------+ |  LBO       |          |    +-------+   +-------+
      | | Native   | |  Traffic   |          |    |IMS APN|   | Other |
      | |          | |            |          |    |  PGW  |   |  APN  |
      | | Client   +-------------------+     |    |       |   |  PGW  |
      | |          | |            |    |     |    +-------+   +-------+
      | +----------+ |            |    |     |         |          |
      +--------------+            +----------+         |          |
                                       |               |          |
                                       |           +-------+   +-------+
                                       |           |  IMS  |   |  App  |
                                       |           |       |   | Server|
                                       v           +-------+   +-------+
```

Figure 2: Untrusted Wi-Fi to Packet Core Integration Model for Wi-Fi Calling

The Wi-Fi calling client implementation uses the ePDG client for IMS APN while the default PDN or Internet APN traffic is locally offloaded (Local Breakout LBO) into the Wi-Fi access network. The "untrusted Wi-Fi" architecture supports multiple APN over SWu, allowing the MNO to also route specific applications traffic associated with one or more APN through the Packet Core, in addition to the IMS APN, if required.

#### 4.1.1.1. IPSec Tunnel Negotation

The IPSec tunnel from the UE to the ePDG is negotiated using IKEv2. The parameters for tunnel negotation in Wi-Fi Calling are as follows:

o The Initiator Identifier (IDi) will be in ID_RFC822_ADDR (email address) form, and be based on the UE’s IMSI@Realm.

- o The Responder Identifier (IDr) will be in ID_FQDN form, and be the APN name that the tunnel should access through the ePDG.

- o EAP should be used for mutual authentication. When on a device with a SIM card, EAP-AKA should be used. On other devices, EAP-TLS is preferred. EAP-Only authentication (in which the server certificate is not sent in an CERT payload) may be used to reduce packet size, but only with mutually authenticating EAP types such as EAP-AKA or EAP-TLS.

- o Strong encryption and authentication algorithms should be used, such as ENCR_AES_CBC, PRF_HMAC_SHA2_256, AUTH_HMAC_SHA2_256_128, and Diffie-Hellman Group 14.

- o The Configuration Request should specify an IPv4 or IPv6 addresses used for handover. The UE may also request ePDG-specific attributes such as P_CSCF_IP4_ADDRESS and P_CSCF_IP6_ADDRESS.

#### 4.1.2. Hybrid Model

3GPP TS 23.402 also defines the concept of "trusted Wi-Fi" architecture, providing another method to integrate with the packet core. The trustworthiness of an access network itself is left to the MNO to decide, but it generally relies on some level of control by the MNO of the Wi-Fi access network either in a direct or indirect manner. One of the key characteristics of the "Trusted Wi-Fi" architecture as defined in 3GPP Release 11, is the client-less approach to support the packet core integration. This solution lacked the support for multiple APNs signaling for the UE when over the Wi-Fi access network, therefore all Wi-Fi offloaded traffic was assumed to be part of the default PDN or Internet APN. With this limitation, Wi-Fi calling cannot be supported as it require its own IMS APN. The hybrid architecture proposed here combines the 3GPP release 11 "trusted Wi-Fi" architecture, with the ePDG based untrusted Wi-Fi architecture. This hybrid model simultaneously supports IMS and other applications specific APNs using the untrusted Wi-Fi model, with the TWAG selectively offloading their traffic, while using the S2a interface for all other default PDN traffic toward the default PGW. This Architecture model is illustrated in figure 3 below

```
                                                  +------+   +----------+
                                                  | IMS  |   |  Other   |
                                                  | Core |   |Services  |
                                                  +------+   +----------+
                                                     |            |
                                                  +------+   +--------+
       +--------------+      +-----------+        |IMS   |   | Other  |
       | +----------+ |      |           |        |P-GW  |   | P-GW   |
       | |   SWu    | |      | +--------+|        +------+   +--------+
       | |  Client  | |      | |SIPTO   ||           |            |
       | |          | |      | ++NAT    ||           |  +--------+
       | +----------+ |      | +--------+|           |  |
       |              |      |           |        +------+
       |     UE       +------+   TWAG    |  S2b   | ePDG |
       |              |      |           +--------+      |
       | +----------+ |      |           |        +------+
       | | Native   | |      |           |
       | | Client   | |      |           |
       | |          | |      |           |  S2a   +--------+
       | +----------+ |      |           +--------+Default|
       +--------------+      +-----------+        | PGW    |
                                                  +--------+
                                                    |
                                                    |
                                                    +
                                                 XXXXXXXX
                                                XX      XX
                                                X        X
                                                XInternetX
                                                X        X
                                                XX      XX
                                                 XXXXXXX
```

Figure 3: Hybrid Wi-Fi to Packet Core integration model for Wi-Fi calling

### 4.1.3. Trusted Model

Enhancements introduced in 3GPP release 12 SaMOG specifications provides the ability to support multiple APN over Wi-Fi access making the support of Wi-Fi calling, and other applications specific APNs possible without the need for IPSec connectivity between the UE and the Packet core. This Architecture model is illustrated in figure 4 below

```
                                                             +-------+
                                                             |  IMS  |
                                                             | Core  |
            +-------------+              +-----------+       +-------+
            |             |              |           |           |
            | +--------+  | Flow mapped  |           |           |
            | |IMS APN |  | to VMAC-01   |           |       +-------+
            | |        +-----------------+           |       |IMS APN|
            | |Client  |  |              |           +-------+ P-GW  |
            | +--------+  |              |           |       +-------+
            |             |              |           |
            |             |              |Release 12 |
            |             |              |  TWAG     |
            |             |              |           |       +-------+
            |             |              |           |       |Default|
            |             |              |           +-------+ APN   |
            | +--------+  | Flow mapped  |           |       | P-GW  |
            | |Default |  | to VMAC-02   |           |       +-------+
            | | APN    +-----------------+           |           |
            | |Client  |  |              |           |           |
            | +--------+  |              |           |           |
            +-------------+              +-----------+           |
                                                               XXXXXX
                                                             XX     XXX
                                                            XX        XX
                                                           X           X
                                                          X            X
                                                          X Internet   X
                                                          X            X
                                                          X           XX
                                                           X         XX
                                                            XX    XXXX
                                                             XXXXXX
```

Figure 4: Trusted Wi-Fi to Packet Core integration model for Wi-Fi calling

#### 4.1.4. Model Selection Criteria

Each of the Wi-Fi to Packet Core Architecture models described in the previous sections comes with its own pros and cons. And selection of a specific architecture model depends on several factors. Some of these factors, which can help determine the appropriate model, are listed below:

*Wi-Fi Access Network Ownership: There are several ownership models available when it comes to Wi-Fi to packet core integration. Wi-Fi Access network may be deployed by the MNO to leverage as another RAT to complement 3G and LTE. Alternatively the Mobile Network Operator may deploy a Managed Wi-Fi network for the Enterprise and SMB customers. The MNO managed Wi-Fi footprint is only portion of the overall Wi-Fi deployment. Third parties such as broadband service providers today own a significant portion of the Wi-Fi access network. For third party owned Wi-Fi access, the Mobile Network Operator may or may not have a direct roaming partnership with the Wi-Fi operator. The ownership model influences the choice of packet core integration architecture.

*Backhaul Network Ownership: From the context of this discussion here, the backhaul refers to the connectivity between WLAN Access network and the Packet core. It consists of a combination of wired access network of the hotspot, Broadband access last mile, Wi-Fi operator core network, Internet etc. These connectivity aspects will be deciding factor for the choice of Wi-Fi packet integration model. For example, Wi-Fi access network may be owned and or operated by the MNO, but if the backhaul involved a third party connection or Internet where MNO does not have control over security and QoS, an untrusted packet core integration may be the viable solution.

*Mobile Offload Requirements: Choice of the Wi-Fi to packet core integration model is not only influenced by voice offload but data offload as well. The untrusted Wi-Fi and the hybrid architectures do support a flexible offload model, allowing the Mobile Network Operator to choose which traffic to backhaul to the Mobile Packet Core to provide charging and added value services, while also leveraging local breakout capabilities on the device. Using the untrusted, and when applicable, the hybrid models allow the Mobile Network Operator to leverage their deployed network architecture for Wi-Fi calling. This makes both the hybrid and the untrusted Wi-Fi architectures valid options to consider depending on the Wi-Fi network ownership requirements.

*Device Capabilities: This greatly influences the choice of Wi-Fi to packet core integration. For example, a trusted approach with multiple PDN support requires the capability on the device to comply with 3GPP release 12 SaMOG enhancements, while the untrusted or hybrid model can leverage existing implementations and do provide a similar level of functionality.

*Support of Non-SIM devices: The MNO can provide value-added services, including voice services on Non-SIM devices The Untrusted Wi-Fi architecture is compatible with Non-SIM devices and provide the same capabilities to these devices as for the SIM devices.

*Network Readiness: This is another influencing factor for the choice of the trust model, as there are dependencies on the Packet Core network elements as well as Wi-Fi access network for the implementation of these models.

## 5. Subscriber Onboarding into Wi-Fi Access Network

Subscriber onboarding into a Wi-Fi access network is the process of getting connected to a WLAN access network and be able to offload mobile traffic successfully. In order to provide a seamless end user experience for Wi-Fi calling, the handset should be able to get connected to the WLAN with minimum or no user interaction. A seamless WLAN onboarding is critical for the smooth hand off of the voice call from LTE to Wi-Fi. There are several factors, which can influence the Wi-Fi onboarding experience. Proper choice of the available deployment options can ensure the subscriber onboarding experience is quite seamless.

### 5.1. Authentication and Identity Management

Before the UE device can successfully get associated with a WLAN access network it needs to get authenticated with the WLAN network. There are several types of user authentication options in use such as Web Portal based authentication, EAP-TTLS, EAP-TLS, EAP-SIM, EAP-AKA etc. Choice of the authentication mechanism depends up on the deployment preferences of the Wi-Fi operator. Web portal based authentication relies on an Open SSID configuration. Once the portal has successfully authenticated the UE device, the traffic is carried over the WLAN air interface without any encryption. EAP authentication mechanisms relies on secured SSIDs mandate the 802.11i based air encryption of the subscriber data in the WLAN access network.

In order to support Wi-Fi calling, one of the EAP based mechanisms will be preferred over the web portal based authentication. In the case of Web based authentication, the user needs to manually enter the username and password credentials or in some cases sign up for a service via Operator portal. But with any of the EAP methods, once the credentials have been established on the UE device, then authentication happens automatically without user intervention and greatly improves the onboarding experience.

If the Wi-Fi operator decides to use a secured SSID for subscriber authentication, choice of the EAP method depends up on the business model. A Standalone Wi-Fi operator may need to rely on non-SIM based EAP authentication mechanisms such as EAP-TTLS or EAP-TLS for their home subscribers. A Wi-Fi operator who has a roaming partnership with an MNO could allow the uSIM credentials of the MNO subscriber to be used for the access. In this case, the Wi-Fi operator will act as a proxy and authenticate the customer credentials with the MNO HSS.

Identity management deals with establishing subscriber identity and associated credentials on the UE device for WLAN onboarding. Identity management and authentication goes hand in hand. Option leverages the same set of identity and credentials (unified identity) for WLAN onboarding and packet core connectivity will simplify the identity management for Wi-Fi calling. However this requires that the WLAN access network is either owned by the MNO or by their roaming partner. With unified identity, typically uSIM credentials will be leveraged for both WLAN onboarding as well as packet core connectivity for SIM devices, and an EAP method used for Non-SIM devices.

## 5.2. Hotspot 2.0 for Seamless Onboarding

Ability for a handset to Seamlessly get connected to WLAN access network is one of the key factors which will influence the overall subscriber experience with Wi-Fi calling. Passpoint specifications defined by the Wi-Fi alliance under the Hotspot 2.0 program supports automatic discovery, selection and onboarding of Wi-Fi clients on to a compatible Wi-Fi access network. Figure-5 below is used to illustrate the hotspot 2.0 solution at a high level:

```
                              +----------------+
                              | Wi-Fi Operator |
                              |  AAA Server /  |
                              |  AAA Proxy     |
                              |                |
                              +----------------+
                                      |
                                      |
                                      |
                              +----------------+              XXXXX
           +-----------+      |                |            XXX   XXX
           |           |      |  +----------+  |           XX       XX
           |           |      |  |  ANQP    |  |          XX         XXX
           |    UE     |      |  | Server   |  |          X            XX
           |           |      |  |          |  |         XX   Roaming   X
           | +-------+ +------+  +----------+  +---------XXInterconnectXX
           | |  ANQP | |      |                |         XX            X
           | | Client| |      |                |          XX          XX
           | |       | |      |  +----------+  |           XX       XXX
           | +-------+ |      |  |  AP/AC   |  |            XX     XX
           +-----------+      |  |          |  |             XXX XXX
                              |  +----------+  |               X+X
                              +----------------+                |
                                      |                         |
                                      |                         |
                                      |                   +------------+
                                    XXXXX                 |   MNO      |
                                  XX    XXX               | AAA Server |
                                 X        XX              |            |
                               XX          XX             +------------+
                               X  External  X                   |
                              XX  Network   XX                  |
                              X    Access    X                  |
                              XX            XX            +------------+
                               XX          XX             |    HSS     |
                                XX       XXX              |            |
                                 XXXX  XXX                +------------+
                                    XXXX
```

Figure 5: Hotspot 2.0 with Service Provider Roaming

ANQP server is the component, which assists with the automatic discovery of WLAN network resources by the UE device.  ANQP server is typically collocated on the Access Point (AP) or the Access Controller (AC). A Hotspot 2.0 compatible UE device will have a built in ANQP client. When a UE roams into the coverage area of a Hotspot 2.0 enabled network, it automatically learns about the network capability via Beacon or Probe Response. Then UE requests a set of network and service level information from the WLAN network. Based up on the info UE can decide which WLAN access is the most preferred and the type credentials it can use for getting connected.

### 5.2.1. Hotspot 2.0 Inter-Operator Roaming for Wi-Fi Calling

MNOs can enter into roaming partnership, which will allow Wi-Fi calling clients to automatically get connected to the WLAN access. This also allows the devices to leverage uSIM credentials or EAP credentials for Non-SIM devices for getting authenticated with the WLAN network. The Wi-Fi operator AAA will function as a proxy in this case and completes the authentication by interfacing with the MNO AAA Server and HSS, for EAP_SIM/EAP_AKA in the MNO packet core.

## 6. Wi-Fi calling deployment in restrictive networks

The use of IPSec to establish a connection to the ePDG, require that the access network allow IPSec tunnel establishment. But some networks won’t allow IPSec traffic either as a security policy or as a side-effect of only allowing "web traffic". In addition, many mainly corporate environments do deploy an HTTP proxy which will also prevent the establishment of an IPSec tunnel. Performing changes to these deployments may not always be possible or cost effective for the corporation or the public venues, especially in an "Untrusted Wi-Fi" model without the MNO involvement. In such situations, the mobile device can leverage the IPSec TCP encapsulation as described in draft-pauly-ipsecme-tcp-encaps-04 and in 3GPP TS 24302, which define the encapsulation of IPsec traffic in TCP. The Mobile device shall enable the TCP encapsulation only after failling to establish an IPSec connection to the ePDG. Figure 6 below shows the TCP encapsulation with the use for TLS to traverse a Proxy and reach the ePDG.

```
   +------------+                 +----------+              +-------+
   | +--------+ |  TCP with HTTP  |          |  TCP with TLS   |       |
   | |  SWu   +==================+==========+================+       |
   | | Client  ------------------------------------IPSEC-------       |
   | |        +==================+==========+================+  ePDG |
   | |Proxy   | |   then TLS     |  Proxy   |                |       |
   | |Config  | | through proxy  | Firewall |                +-------+
   | +--------+ |                +----------+                    |
   |    UE      |                                                |
   +------------+                                            +-------+
                                                             |       |
                                                             | IMS   |
                                                             | PGW   |
                                                             +-------+
```

Figure 6: Use of TCP encapsulation for IPSec

When an HTTP proxy is deployed, the UE should connect to the eDPG through the proxy and then establish a TLS connection toward the ePDG. TLS is not used for securing the link, but to traverse the HTTP Proxy, and is configured with NULL-Cipher. This model allows Wi-Fi calling to operate even in restrictive networks.

## 7. RF Network Performance Optimization

Quality of the Wi-Fi calling experience would be as good or as bad as Radio network itself. Three network performance KPIs which impact the quality of voice are latency, jitter and packet drops. A healthy network is critical to ensure that these KPIs will meet the thresholds allowed to meet the acceptable voice quality. This section primarily talks about various performance optimization mechanisms available on the Wi-Fi Radio network.

### 7.1. Radio Resource Management

Radio Resource Management (RRM) aka Wi-Fi SON refers to the co-ordinated fine-tuning of the various RF network parameters among access points connected in a Wi-Fi network. It is very typical for Wi-Fi deployments from multiple operators to co-exist in the same hotspot. Scope RF fine tuning will be limited to the access points which are managed by the same operator in a specific hotspot. RRM fine-tuning will be typically performed by a centralized entity such as Access Controller. Some deployments which may not leverage AC such as Residential Gateways could leverage a cloud based RRM or SON Server. RRM controller continuously analyze the existing RF environment automatically adjust the power and channel configurations of access points to help mitigate issues such as co-channel interface and signal coverage. A proper implementation of RRM can greatly influence the RF performance and will have a positive impact on network KPIs that influence the Wi-Fi calling experience.

### 7.2. Wi-Fi Roaming Optimization

Roaming from the context of the discussion here refers to the hand of of a UE device from one Access Point to another Access Point in the same Extended Services Set (ESS) or mobility domain. Unlike cellular roaming between base stations, which is initiated by the network, in Wi-Fi the roaming is initiated by the UE device. A UE typically decides to disconnect from the current access point when some of the RF measurements such as RSSI, SNR etc. drops below certain threshold. There are other APs in the range with acceptable measurements the UE will start re-association process with one of the target APs. End user experience for a Wi-Fi call, which is active at the time of the hand off, will depend up on multiple factors. One critical factor is the time taken for the UE traffic to resume during the hand off. Also it is important that UE is able to make the optimum selection of the target AP from the list of available APs in the range. Discussed below are few IEEE 802.11 based mechanisms available to optimize the roaming.

#### 7.2.1. Fast BSS Transition

IEEE 802.11r based fast BSS transition (FT) helps reduce the handoff time for a UE when it roams from one AP to another with in an ESS, which is enabled, with an EAP based authentication. Without FT, the UE will have to go through the full authentication process with the RADIUS server and device fresh set of encryption for 802.11i air encryption. When FT is enabled, the client will have an initial handshake with the target AP while still connected to the original AP. This handshake allows client and target APs to derive the encryption keys in advance to reduce the hand off time. Fast Transition can significantly improve the end user experience for the voice calls, which are active during a hand off.

#### 7.2.2. 802.11k based Neighbor Reports

IEEE 802.11k enhancements allow a UE device to request from the current AP to which it is connected for a recommended list of neighboring APs for roaming. Up on receiving the client request, the AP responds with a list of neighbors on the same WLAN with the Wi-Fi channel numbers. Neighbor list is created by the AP based up on the Radio Resource Measurements and includes the best potential roaming targets for the UE. Neighbor list allows UE to reduce the scanning time when it is time to roam into a new AP in the same WLAN and there by improves the roaming performance. It is recommended to enable

802.11k along with Fast BSS transmission for optimum roaming performance.

#### 7.2.3. 802.11v based Assisted Roaming and Load Balancing

Typical WLAN deployments will have APs with overlapping coverage areas. This is done on purpose to seamless handoff and also to address capacity requirements. Load distribution of UEs in the same coverage area may be helpful to proactively manage the bandwidth requirements and there by improve the subscriber experience. In the most rudimentary form, some of the load balancing solutions relies on the brute force method of ignoring the association requests from a UE by the APs with high load. Another more sophisticated mechanism is to leverage 802.11v based network assisted roaming. 802.11v allows unsolicited BSS transmission management messages from AP towards the client with a list of preferred APs to make roaming decisions. If the AP is experiencing high load, or bad connectivity from the client it may send an unsolicited BSS transmission management frame with the recommended list of APs to roam into. Depending up on the client implementation, it may or may not honor this info while making oaming decisions.

## 8. QoS Deployment Considerations for Wi-Fi Calling

This section covers the traffic prioritization mechanisms available in various segments of the overall traffic path of the Wi-Fi calling signaling and bearer sessions. Flexibility control of the QoS implementations will depend up on various factors such as ownership and management of the WLAN access network, Wi-Fi to packet core integration model etc.

### 8.1. Wi-Fi Access Network QoS

Traffic prioritization in the WLAN for Carrier Wi-Fi calls can be achieved by implementing Wi-Fi Multimedia (WMM). WMM consists of a subset of IEEE 802.11e enhancements for Wi-Fi. WMM defines four Access Categories, AC1, AC2, AC3 and AC4. AC1 is mapped against voice, AC2 is mapped against video, AC3 is mapped against best effort traffic and AC3 is mapped against Background traffic. Each of these Access Categories is mapped against one or more 802.11e User Priority (UP) values. UP has range from 0 to 7. Higher UP values typically gets more expedited over the air treatment EDCA mechanism for channel access defined in 802.11e is modified to make sure that traffic in higher UP queues get higher priority treatment. WMM can only leveraged if the client can do the right classification and Access points also support it.

## 8.2. End to End QoS

While QoS on the WLAN access network is critical, that by it may not be sufficient to maintain the subscriber quality of experience. It is important to enable QoS prioritization across all the network segments, which form part of the end-to-end voice path. Flexibility of the QoS implementation along the network segments will depend up on the trust models, which are discussed earlier. For example, if the transit path between WLAN network and Packet Core is include Internet, no QoS prioritization can be implemented over the Internet backhaul. How ever for deployment scenarios in which all network segments along the voice traffic path are managed either by the Mobile operator or their partners, then it makes much easier to implement end to end QoS. End to end QoS Classification for Wi-Fi calling is illustrated in figure 7 below.

```
                   Voice UP 6          Voice DSCP 46
                 Voice Sig. UP 4      Voice Sig. DSCP 24

                +--------------+     +----------------+
                | WMM or WMM+AC|     | DiffServ QoS   |
                |              |     |                |
                v              v     v                v

            +----+           +---------+          +---------+
            |    |           |  WLAN   |          |         |
            |    |           |         |          |         |
            | UE |           | +----+  |          |  TWAG/  |
            |    +-----------+ | AP/|  +----------+  ePDG   |  <------+
            |    |           | | AC |  |          |         |         |
            |    |           | +----+  |          |         |         |
            +----+           +---------+          +---------+         |
                                                       |  Dedictated |
                                           Voice QCI 1|  and         |
                                           Sig. QCI 5 |  Default     |
                                                       |  Bearers    |
                                                       |             |
                                                  +---------+  <------+
                                                  |         |
                                                  |  P+GW   |
                                                  |         |  <------+
                                                  +---------+         |
                                                       |              |
                                                       |    DiffSrv   |
                                                       |      QoS     |
                                                       |              |
                                                  +---------+         |
                                                  |         |         |
                                                  |  IMS    |  <------+
                                                  |         |
                                                  +---------+
```

Figure 7: End-to-end QoS Reference Model

This QOS reference model assumes that, MNO or their roaming partners manage all the segments in the end-to-end path for voice signaling and voice bearer traffic.  Model also assumes that transit path between WLAN and Packet core is private and secured and does not traverse Internet.

QoS reference model leverages WLAN access network leverages WMM that is described in the previous section, UP value of 6 is typically used for voice bearer traffic and UP value of 4 is used for voice signaling traffic. In order for voice to get the proper prioritization, WMM needs to be supported and enabled on both UE and the WLAN network.

In the transit IP network between WLAN and packet core, DSCP based QoS prioritization can be deployed if the connectivity is part of a managed transport. DSCP value of 46 is typically used for marking voice bearer and DSCP value of 24 is typically used for marking voice signaling. Proper traffic prioritization will depend up on whether DiffSrv QoS is enabled in the transit network.

Between P-GW and ePDG or TWAG, dedicated bearer with QCI value 1 will be established dynamically for voice calls. For signaling traffic a default bearer with QCI value of 5 will be used. These QCI values are mapped against specific QoS SLAs and allocation retention policies (ARP).

# 9. Wi-Fi Calling Client Considerations

Wi-Fi Calling client device functionality requirements depend on the on the models used for WLAN to packet core integration. At a minimum the clients should support IMS User Agent as defined in the 3GPP spec and be able to send and receive both IMS signaling and bearer traffic over a Wi-Fi access point. In addition, an SWu client that supports IPSec will can use ePDG-based packet core integration. This section talks about some of the client side implementation considerations for Wi-Fi calling.

## 9.1. Access Selection Criteria

The client device must select which RAT (cellular or Wi-Fi) it will use for communication to the cellular network. Commonly deployed access selection criteria is described below:

Device Local Policy Profile: In this case, the logic is defined by locally configured policy. Local policy may allow the end user to set prereferences. It is also possible for carriers to push these profiles to the device. Some MNOs may prefer cellular instead of Wi-Fi for voice service when both RAT technologies are available. Some other carriers may have Wi-Fi preferred approach for IMS APN when both RAT technologies are available. If Passpoint is enabled on the Wi-Fi access network, the client may take into account network loading conditions learned from the ANQP server to decide whether to offload IMS traffic into the Wi-Fi network.

## 9.2. Inter-RAT Handover

Inter-RAT handover refers to the handover of an active voice call without service disruption when the UE switches out from one RAT technology to another. Implementations must support handovers between Wi-Fi and LTE.

Handover between LTE and Wi-Fi is acheived by maintaining IP or IPv6 addresses between the LTE interface and the IPSec tunnel over Wi-Fi. If the IPSec tunnel is negotiated while a call is already in progress, the IKEv2 Configuration Request should specify the local address of the LTE interface in order to get assigned the same address on the IPSec tunnel. Similarly, handover from an IPSec tunnel over Wi-Fi to LTE requires the LTE interface to be brought up with the same address as the tunnel. Maintaining the address allows the client to not interrupt TCP or UDP connections that are using the local address for communication. In a system that uses POSIX sockets, for example, the handover must be done in such a way that the sockets do not need to be closed and re-opened.

## 9.3. MTU Considerations

When handing over between LTE and IPSec tunnels over Wi-Fi, the client device should be aware of the Maximum Transmission Unit (MTU) of each interface. It is possible that the effective MTU for the IPSec tunnel (which can be calculated as the MTU of the Wi-Fi interface minus the overhead for ESP encryption) is notably smaller than the effective MTU of the LTE interface. For UDP flows, they should avoid sending large datagrams that could get fragmented when handing over between RATs. For TCP flows, the Maximum Segment Size based on the MTU SHOULD be re-calculated upon handover.

## 9.4. Congestion Management

Radio Network Performance management and QoS considerations described earlier can significantly contribute to the overall QoE for Wi-Fi calling. A client driven congestion management mechanism can positively augment the overall experience. The idea is to dynamically change the bandwidth requirements for the call based up on the network congestion conditions. Network resource requirements (bandwidth, packets per second etc.) per call are directly proportional to the type of codec and the packetization rate. Sometimes it may be desirable to switch out to a lower audio codec to keep the drop, delay and jitter characteristics under acceptable levels during periods of network congestion. Explicit Congestion Notification for RTP over UDP defined in RFC 6679 can be used to inform network congestion to the end clients. But this requires the network elements to mark the ECN bits on the IP header of the packet when congestion conditions are encountered.

## 10. Acknowledgements

Authors would like to acknowledge the inputs and advice provided by Eduardo Abrantes and Ajoy Singh.

## 11. Informative References

[RFC4066] Liebsch, M., Ed., Singh, A., Ed., Chaskar, H., Funato, D., and E. Shim, "Candidate Access Router Discovery (CARD)", RFC 4066, DOI 10.17487/RFC4066, July 2005, <http://www.rfc-editor.org/info/rfc4066>.

[RFC4187] Arkko, J. and H. Haverinen, "Extensible Authentication Protocol Method for 3rd Generation Authentication and Key Agreement (EAP-AKA)", RFC 4187, DOI 10.17487/RFC4187, January 2006, <http://www.rfc-editor.org/info/rfc4187>.

[RFC4555] Eronen, P., "IKEv2 Mobility and Multihoming Protocol (MOBIKE)", RFC 4555, DOI 10.17487/RFC4555, June 2006, <http://www.rfc-editor.org/info/rfc4555>.

[RFC4881] El Malki, K., Ed., "Low-Latency Handoffs in Mobile IPv4", RFC 4881, DOI 10.17487/RFC4881, June 2007, <http://www.rfc-editor.org/info/rfc4881>.

[RFC5213] Gundavelli, S., Ed., Leung, K., Devarapalli, V., Chowdhury, K., and B. Patil, "Proxy Mobile IPv6", RFC 5213, DOI 10.17487/RFC5213, August 2008, <http://www.rfc-editor.org/info/rfc5213>.

[RFC5568] Koodli, R., Ed., "Mobile IPv6 Fast Handovers", RFC 5568, DOI 10.17487/RFC5568, July 2009, <http://www.rfc-editor.org/info/rfc5568>.

[RFC5944] Perkins, C., Ed., "IP Mobility Support for IPv4, Revised", RFC 5944, DOI 10.17487/RFC5944, November 2010, <http://www.rfc-editor.org/info/rfc5944>.

[RFC6275] Perkins, C., Ed., Johnson, D., and J. Arkko, "Mobility Support in IPv6", RFC 6275, DOI 10.17487/RFC6275, July 2011, <http://www.rfc-editor.org/info/rfc6275>.

[TS23402] "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Architecture Enhancements for non-3GPP Accesses.", 2009.

Authors' Addresses

Byju Pularikkal
Cisco Systems
170 West Tasman Drive
San Jose
United States

Email: byjupg@cisco.com

Tommy Pauly
Apple Inc.
1 Infinite Loop
Cupertino, California  95014
US

Email: tpauly@apple.com

Mark Grayson
Cisco Systems
10 New Square Park
Feltham
United Kingdom

Email: mgrayson@cisco.com

Sri Gundavelli
Cisco Systems
170 West Tasman Drive
San Jose
United States

Email: sgundave@cisco.com

Samy Touati
Ericsson
300 Holger Way
San Jose, California  95134
US

Email: samy.touati@ericsson.com

Operations and Management Area Work Group
Internet-Draft
Intended status: Informational
Expires: September 27, 2017

S. Rich
Cisco Systems

T. Dahm
Google

27 March 2017

# MUD Lifecyle: A Manufacturer’s Perspective
draft-srich-opsawg-mud-manu-lifecycle-01.txt

## Abstract

Manufacturer Usage Descriptions, or MUDs, allow a manufacturer to cheaply and simply describe to the network the accesses required by an IoT device without adding any extra cost or software to the devices themselves. By doing so, the network infrastructure devices can apply access policies automatically which increase the overall security of the entire network, not just for the IoT devices themselves. This document describes the lifecycle of Manufacturer Usage Descriptions (MUDs) by describing detailed MUD scenarios from the perspective of device manufacturers.

## Status of This Memo

## 1. Introduction

The addition of IoT devices to a network expands the attack surface of that network. Even if a device does not have exploitable vulnerabilities (in the sense of an attacker injecting and running malware on it), it may be susceptible to denial-of-service (DoS) attacks and thus could have its functionality impaired by attackers. Recent events have shown just how real, and not just theoretical, such attacks can be.

A detailed summary of the current state of understanding of the Mirai botnet’s use of IoT devices can be found in [MIRAI]. It is estimated that around 100,000 IoT devices generated more than a terabit per second of DDoS traffic.

Also consider the Sony Cameras IP Security article [SONYCAMS] which describes a vulnerability in many camera models which could be exploited to launch attacks like those seen in the massive DDoS attack on DynDNS in [DynDNS]. As both of these incidents show, more network-accessible devices which can connect to arbitrary external addresses can, if those devices permit too much access or if they have vulnerabilities which allow arbitrary code execution, be used by attackers to amplify attacks and to do so by using origin addresses spanning broad ranges of networks.

Concerns about the negative possibilities of attacks related to IoT devices is also discussed in [MITTECH] that also discusses some of the regulatory and government angles in play. In a recent move described in [USGSUIT], the U.S. Federal Government has taken the step of suing D-Link, accusing it of ‘‘poor security practices’’ for some of its IoT devices.

MUD provides a light-weight model of achieving very effective baseline security for IoT devices by simply allowing a network to automatically configure the required network access for IoT devices so that they can perform their intended functions without granting them gratuitous, unrestricted network privilege.

## 2. MUD High-level Introduction

Manufacturer Usage Descriptions (MUDs) provide advice to end networks on how to treat specific classes of devices. The MUD architecture is explained in [LEAR2017], but we will describe it briefly here and also discuss details where necessary to understand this document. At its most basic, MUD is a system by which the IoT device itself tells the network exactly how to retrieve its network access requirements (in a ‘‘MUD File’’, which is the term used in the MUD specification to refer to the file which contains the description of an IoT device’s network access requirements), and network infrastructure can fetch and act upon this information. The MUD File itself is a static text file which the network infrastructure element responsible for it can retrieve from the manufacturer or from whomever the manufacturer delegates the responsibility to. The MUD file may be cached, so when served, the MUD file should be returned with a ‘‘max-age’’ value which lets the requestor know how long it can cache it.

To add MUD support to an IoT device is a very minimal change: add the URL for the MUD File as the ‘‘MUD URI’’ to whatever dynamic network registration protocol which is currently being used by the device (e.g. DHCP, etc.). It is so simple that the device manufacturer can statically compile the URI into the firmware of the device. The essential point is that MUD does not force a large behavioral change on the IoT device itself, and the serving up of the MUD file during the lifetime of the devices is similarly relatively low-impact. The bulk of the complexity of MUD is concentrated within the network elements which perform operations to retrieve the MUD files, possibly cache them, and then configure the network in response, but even there, the network elements effected mostly already perform all of these actions, albeit not automatically in most cases.

For this description, one can consider three general classes of actors in the MUD ecosystem:

- Device manufacturers

- Networking equipment manufacturers

- o Network operators

Note that end users are not mentioned here, as their involvement in MUD is minimal at best (and likely only present in the simplest of deployments). Note also that ‘‘Device manufacturers’’ are described with the assumption that they will both include MUD URIs within their devices as well as service MUD URL requests (via a cloud service or via their own web infrastructures). It is possible that a manufacturer will delegate the MUD URL retrieval function to a third party. The question of who actually services network requests for the MUD URL is an administrative one and does not affect the MUD architecture. It does give device manufactures more flexibility, though, in managing their investment into the MUD ecosystem.

This document will describe the MUD ‘‘lifecycle’’ from the standpoint of manufacturers, but it is also intended to be informative to persons interested in standardization, installation, or other areas where MUD may be in play. Where appropriate, suggestions of best practices will be given if there are no specific hard requirements.

## 3. Terminology

Before going into descriptions how MUD works, we will list terms used within the MUD ecosystem:

MUD
: Manufacturer Usage Description

MUD file
: a file containing YANG-based JSON that describes a recommended behavior

MUD file server
: an HTTPS server that hosts a MUD file

MUD controller
: the system that requests and receives the MUD file from the MUD server. After it has processed a MUD file it may direct changes to relevant network elements

URL
: Universal Resource Locator

URI
: Universal Resource Identifier. The difference between a ‘‘URI’’ and a ‘‘URL’’ is that a URI is intended to be used as an identifier in a general sense, whereas a URL is a specific use case of a URI that is used to access something at a particular network location

MUD URI
: a URI that an IoT device carries and which will be issued during operations such as DHCP requests which can be used as a URL to retrieve a MUD file

MUD URL
: the MUD URI being used as a URL

IEEE 802.1AR
: A IEEE specification for a certification-based approach for communicating device characteristics

YANG
: A data modeling language for the definition of data sent over the NETCONF network configuration protocol [RFC6020]

NETCONF
: Network Configuration Protocol [RFC6241]

JSON
: Javascript Object Notation, a human- as well as machine-readable file format containing textual representations of ‘‘objects’’ such as strings of characters, numbers, boolean values, and lists and dictionaries of such objects and collections of objects

Many of these terms are in common usage with the IETF or other network standards bodies and are thus used for consistency. More information about terms like ‘‘URL’’, ‘‘URI’’, ‘‘YANG’’, and ‘‘NETCONF’’ can be found in the standards and references published by the IETF and others. The value in distinguishing ‘‘URI’’ and ‘‘URL’’ will hopefully become more apparent when MUD file caching is discussed (during which time, already-retrieved MUD files will be used if the URI lookup returns a match). The actual text of a ‘‘MUD URI’’ and a ‘‘MUD URL’’ will generally be identical; the distinction lies in the use of it by various elements (IoT devices, network devices, and web services).

## 4.  MUD Operation

```
+--------------------------+ +------------------------------------------+
| [Manufacturer]           | | [Customer]                               |
|             -------      | |                                          |
|     +----->(       )  6  | | +------------+  7  +----------------+    |
|     |      |-------|<--------|    MUD     |---->| Network Policy |    |
|     | 2:MUD|       |-------->| Controller |     |   Management   |    |
|     | File (       )     | | +------------+     +----------------+    |
|     |       -------      | |            ^        |                    |
|     |                    | |   5:MUD    |        | 8                  |
|  (1:Create Device)       | |     URI    |        v                    |
|         |                | |          +-------------+                 |
|         |                | |          |  (Switch)   |                 |
+---------|----------------+ |          +-------------+                 |
          v                  |           |  |  |  |  |                  |
+-------------------+ 3:Buy  | 4:Deploy  |  |  |  |  |                  |
| Distribution    ---------->|           X  X  X  X  X  ...             |
| Channels          |        |                                          |
+-------------------+        +------------------------------------------+
```

    6: MUD URI used as URL to request MUD File
    7: MUD Controller informs network policy engine about ACLs
    8: Network policy applied as close to IoT device as possible

Figure 1: MUD-related network information flow

A full description of MUD is given in [LEAR2017].  In short, when a device such as an IP-enabled lightbulb is connected to the network and given power, that device will perform some action to acquire a network identity, including an IP address, such as by making a DHCP request.  If that request has a MUD URI in it, equipment in the network (not necessarily the DHCP server) can use that URI to retrieve the device's MUD file from the MUD file server.  Some other networking component (the switch to which the bulb in connected, for example) can then act on the contents of the retrieved MUD file and apply the appropriate configurations to allow the device to function normally while restricting where it can connect.

A MUD file's contents will mostly contain descriptions of which protocols are required by the device and over what port or ports.

From the perspective of a manufacturer, the essential elements to note are the following:

1. On the device itself, the only change required to add MUD compliance/functionality is to add a field populated with a URI to whatever network access protocol is already being used (i.e., DHCP, IPv6 AD, etc.). This will be a static text string which will probably remain constant throughout the life of the product and which is identical for every instance of a product run (i.e., there is no per-serial-number version of the MUD URI)

2. The MUD file which is to be returned via an HTTPS server can be a static file and can be reused for devices which have the same network access requirements. The service which returns the MUD file will not be responsible for any security policy enforcement, as that is the job of the network which contains the devices themselves

3. MUD files are fairly short (on the order of tens of lines of text) and are thus trivial to serve either directly and are amenable to caching

4. The act of retrieving the MUD file and of acting on it is entirely up to the network infrastructure and not a responsibility of the IoT devices themselves. MUD does not impose any behavioral requirements on the IoT devices themselves other than that they must send the MUD URI during network access configuration, as mentioned earlier

How does MUD work in practice? Figure 1 shows a representation of the high-level MUD information flow. This document deals almost exclusively with elements in the upper left of that figure. Specifically, it describes what a manufacturer should do to put a MUD file into a device and what is required for a manufacturer (or a designee of the manufacturer) to answer requests for MUD files from network operators whose networks provide connectivity for such devices.

## 5. Device Manufacturer Considerations

The device manufacturers have the most insight into what resources the devices will need once they are installed in a network. They are thus best-suited to author the network profiles which will be required by the devices that they make for correct operation. Conversely, each manufacturer cannot know what each network’s other requirements happen to be. As a result, the manufactures should provide configuration requirements for their devices which network operators can apply in a way best suited for their networks. The network operator can optimize operations through caching, LAN segregation, etc., and can use the MUD information to further secure the network.

If a manufacturer makes many devices which have similar network access requirements, that manufacturer may want to leverage common profiles. They should do so only when the profiles are truly close enough to be treated as the same.

Device manufacturers have three responsibilities under MUD:

- They must author a MUD profile which describes a device’s requirements for network access

- They must encode a MUD URI into the device such that when the device performs DHCP or similar

- The MUD File must be hosted on a publicly-available web server

Since the MUD profiles can be static files, there is very little overhead required to serve these profiles. Due to their static nature, they are inherently cacheable.

Similarly, since the URI can be essentially static (the actual device configurations are easily updatable since they are contained in the MUD file, not the URI), the manufacturer can assign a name space and begin encoding the URIs into the devices relatively early in the manufacturing process, including before the MUD specification is finalized. An important point is that manufacturers should adopt and follow a nomenclature that insures that they can sufficiently distinguish classes or families of devices with different requirements and assign them different URIs. From a security standpoint, it is better to have several URIs with more granular security profiles than it is to have a very few URIs with "catch-all" (and thus more open) security profiles. This ensures that a customer using a single family of devices will have the most closed network configuration possible.

If the device manufacturer decides to update the profile, then it may do so at any time, independently of updates to the firmware on the devices themselves. If it is expected that a profile may change frequently (say, for a new class of devices which aren’t fully understood yet), then the MUD profile for said device should be served with a fairly short max-age (as compared to a device with a well-established network access profile).

## 6. High-level MUD Lifecycle

The following lifecycle description is described considering a single device. As additional devices are added to a portfolio, the same steps are taken for each one where necessary. Each step can be isolated or coordinated with other device instances where convenient. There is little coupling inherent in the way that the various phases of MUD deployment operates to impose strict requirements in this area.

1. Based on a device’s function, a MUD profile is either:

   - Chosen from a library of existing profiles for similar devices

   - Written anew to describe this device’s network requirements

2. If the profile is pre-existing, the a choice is made if this device will receive a new URI or if it should be classed as identical to existing devices and use the same URI

3. The chosen URI is assigned to the device so that when the device performs network initialization, the URI is included in the request (i.e., DHCP, ANIMA, etc.)

4. In parallel or in advance (but prior to first customer shipment), the device manufacturer should allocate in an appropriate namespace and place the MUD profiles for when the URI is used as a URL.

5. The MUD profile should be made available to customers until such a time that the device is unsupported. While it is outside the scope of this document, The manufacturer should support MUD profile retrieval for each device for at least as long as the manufacturer supports the devices themselves.

6. If the profile is found to contain an error, the manufacturer should update the profile. Devices which are already deployed will continue to use the original URI (unless a firmware updates changes it), so the original profile should be corrected

7. If a device manufacturer chooses to update a MUD-enabled device’s firmware, the manufacturer may update the MUD URI to a new one. The manufacturer should change the URI if the network access requirements of the new firmware are sufficiently different from those of the original firmware version.

## 7. MUD URI

The MUD URI is a very visible and important part of MUD that is best done correctly from the start, for once it is embedded in an IoT device, changing it for the fielded devices will be, at best, inconvenient. Choosing a scheme for organizing the ''name space'' for the portion of the URI which is controlled by the device manufacturer may have knock-on effects such as the URL GET request routing behavior that must be supported during MUD file retrieval.

The format of the URI is:

    https://authority/.well-known/mud/mud-rev/model

where ''mud-rev'' is currently the literal string ''v1'', and may be suffixed with '' ?extras ''. Referencing [RFC3986], the authority element is described by the ''authority'' type, the model element by the ''segment'' type, and extras by the ''query'' type. This gives considerable flexibility to manufacturers to structure their various namespaces to handle a huge variety of device types. However, this document will restrict itself to describing a very simple URI encoding scheme.

In the following, we will use ''example.com'' as the authority element. By far, the simplest method of assigning MUD URIs to devices is to assign each distinct model number a URI of the form

    https://example.com/.well-known/mud/v1/model

where the ''model'' element is literally the model number of the device. If a manufacturer has a model number collision problem (possibly because of acquisitions of other companies, for example), a simple scheme of a prefix or a suffix, set off with a hyphen or similar, will suffice to disambiguate them. Since the MUD files are relatively small, there is likely little value in conjuring schemes to save disk space with complicated naming conventions or structure.

## 8. MUD File Serving: Operations, Lifetypes, and Transfer

The previous section discussed how one might design the URI namespace for MUD files. Another very important consideration is the total lifecycle of the serving of MUD files via the internet for an appropriate length of time and what to do if one wants to transfer the responsibility of serving MUD files to some other entity. This section will describe several scenarios and suggest options for the transfer of responsibility of MUD files to other providers. There is no single set policy for these various activities, and organizations are free to decide how and when these transfers occur. There are technical considerations that must be dealt with, but this is not unlike outsourcing subsections of one’s web site to payment partners or other specialists if so desired.

The single largest factor in thinking about serving MUD files throughout their lifetimes is the relative ‘‘permanence’’ of the URI itself (since, for some types of devices, at least, the buried-in URI will be essentially indelible). Even if a device has a more fungible MUD URI (say, because it is easily and frequently updated), it is still wise to consider the case when a device’s MUD URI cannot be easily updated since this represents the most problematic case. Networks containing the MUD-enabled devices will make network requests to retrieve the MUD files. The MUD URIs are, quite literally, the URLs of the MUD files. There, network infrastructure devices from potentially anywhere on the internet will try to retrieve these MUD files. The volume of requests will be simple to handle (given that MUD files are static and small and that MUD servers in the network will be able to cache them and avoid redundant retrievals).

A very simple and direct way to manage MUD files and make the possible future delegation of MUD file serving to a $3^{rd}$-party is to assign a URI DNS ‘‘namespace’’ for your company’s MUD files. For example, using the fictional company ‘‘Acme Lightbulb and Sensor’’ and its web presence at ‘‘https://acmels.com’’, the DNS namespace for MUD files could be

mud.acmels.com

which can serve as the authority section of the MUD URI. If Acme wants to serve the MUD files themselves, then they can provision an HTTPS service that serves that address and return the requested MUD files, or they can create a CNAME to point to the actual entity who will answer the requests.

## 9. Security Considerations

The bulk of this document describes the use of MUD to increase the security of a network. However, it is possible to compromise the effectiveness of MUD by attacking its behavior directly. This section discusses the known attacks and describes possible mitigations (all from the manufacturer’s perspective). This section also attempts to clarify the limits to which MUD is expected to perform in terms of increasing security.

The first and most obvious attack scenario is that a malicious or compromised device can issue a MUD URI which allows that device to communicate too permissively, either by having the URI refer to an unintended file or by simply putting too permissive a set of rules in the otherwise-legitimate MUD File. A manufacturer SHOULD employ secure development best practices to take reasonable steps to insure that their devices behave correctly at least up to the point that they are shipped and that their web services follow all BCPs.

Other attacks are not manufacturer-specific and will not be covered in this document. They will instead be discussed in TBD which focuses on the network operator's perspective of MUD.

## 10. IANA Considerations

This document has no actions for IANA.

## 11. Normative References

[LEAR2017]
Lear, E., "Manufacturer Usage Description Specification", draft-ietf-opsawg-mud-03, January 05, 2017

[WEIS2017]
Weis, B., "RADIUS Extensions for Manufacturer Usage Description", draft-weis-radext-mud-00, October 25, 2016

[RFC6020]
Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", IETF RFC 6020, 2010

[RFC6241]
Enns, R., Bjorklund, M., Schoenwaelder, J. and Bierman, A., "Network Configuration Protocol (NETCONF)", IETF RFC 6241, 2011

[RFC3986]
Berners-Less, T., Fielding, R., Masinter, L., "Uniform Resource Identifier (URI): Generic Syntax", IETF RFC 3986, 2005

## 12. Informative References

[RFC2882]
Mitton, D., "Network Access Servers Requirements: Extended RADIUS

Practices", RFC2882, July 2000

[MIRAI]
https://www.flashpoint-intel.com/action-analysis-mirai-botnet-attacks-dyn/

[SONYCAMS]
http://www.pcworld.com/article/3147311/security/backdoor-accounts-found-in-80-sony-ip-security-camera-models.html

[DynDNS]
http://www.pcworld.com/article/3134056/hacking/an-iot-botnet-is-partly-behind-fridays-massive-ddos-attack.html

[MITTECH]
https://www.technologyreview.com/s/603015/security-experts-warn-congress-that-the-internet-of-things-could-kill-people/

[USGSUIT]
https://www.cnet.com/news/d-link-lawsuit-ftc-security-hackers/

Authors’ Addresses

Steven Rich
Cisco Systems, Inc.
170 West Tasman Dr.
San Jose, CA 95134

Email: srich@cisco.com

Thorsten Dahm
Google Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043

Email: thorstendlux@google.com

Operations and Management Area Work Group
Internet-Draft
Intended status: Informational
Expires: September 12, 2017

S. Rich
Cisco Systems

T. Dahm
Google

12 March 2017

# MUD Lifecyle: A Network Operator's Perspective
draft-srich-opsawg-mud-net-lifecycle-00.txt

## Abstract

This memo describes the lifecycle of MUD as seen from the perspective of a network operator. It is informational and intended to help provide perspective around the operation of a network which connects MUD-supporting devices and uses MUD-supporting network infrastructure. All phases of network operation that involves or affects MUD will be described. Considerations specific to device manufacturers will be described elsewhere. Considerations relevant to network equipment manufacturers and networking software authors will be described where appropriate where MUD behavior is affected.

## Status of This Memo

# 1. MUD Introduction

Network architects and operators have the goal of designing and operating networks so that they are reliable, secure, and operate correctly. Making them do so requires that the network permit traffic which is intended to be allowed on the network while rejecting or blocking traffic which is not. Both goals are met with a combination of policies and configurations which promote efficient routing of packets for certain classes of traffic and which rate limit or even block (possibly by black-holing) other classes of unwanted or lower-priority traffic.

A common assumption is that devices on the inside of the network can have relatively unrestricted access to other parts of the network and to the local network segment. This is reasonable for devices which themselves have certain configurations which will naturally govern which network access they require. For example, a printer will usually be configured to accept connections from hosts which wish to print to it. The printer itself may not tend to initiate outbound connections and thus does not require a complex set of custom ACLs. If the printer needs external connectivity, the usual scenario is to allow the printer to make outbound connections while still preventing inbound connections using a stateful firewall rule or similar. However, there are often no rules preventing the printer from making arbitrary connections within network delineated by the firewall.

Other devices such as general-purpose end-user hosts (PCs, Mac, etc.) might need unrestricted access, at least in the outbound direction, because, contrary to the printer example, end-user hosts are generally expected to make outbound connections to an unpredictable number of hosts. Even if outbound restrictions to certain ports (such as 80, 443, 22, 25, etc.) are enforced, the destination address may be unrestricted. As stated above, restrictions from internal hosts to internal addresses may be even more lax.

Enter into this situation IoT devices which may be introduced to the network in the thousands and which may have unspecified or unclear requirements for network access. For example, IoT light bulbs may need to talk to DNS, NTP, LLDP, DHCP, and a controller on the local network and nothing else. An IoT thermostat may need to talk to DNS, NTP, LLDP, DHCP, and its cloud-based controller, but nothing else. For both of these cases, while their specific requirements vary, knowing each one’s requirements would allow a tight set of ACLs to be imposed, all the way to the port level, to limit what connectivity is afforded to each individual instance.

Recent examples of IoT-based malware campaigns will not be repeated here and the benefits of providing such security will no doubt be obvious to network operators. What has not been available before MUD is an ability to automatically retrieve configuration policy and then automatically apply it for each device. This document will describe the ‘‘lifecycle’’ of MUD from the perspective of a network operator. The details of the protocol and contents of the MUD file itself are described in [LEAR2017], and familiarity with it is assumed for this document.

## 2. Terminology

This document will use some terms and abbreviations which will be listed and described in this section.

MPD
: "MUD-Protected Device" - While this is a possibly tedious use of a three-letter acronym, repeated use of "MUD-protected device" or similar is equally tedious

AAA Server
: "Authentication, Authorization, and Accounting Server" - A network service which processes AAA requests

ACL
: "Access Control List" - In the context of this document, an ACL will refer specifically to those which are specified in a MUD file and which get applied at some point in the network to enforce the security policy needed by a device. These ACLs may be configured down the port into which the device the is plugged, and they may be applied "dynamically" in the sense that they appear in response

to the MUD request as opposed to a static configuration. They will not be dynamic in the sense that they change frequently. The actual implementation by any particular vendor is left up to that vendor and thus may differ from the examples given in this document.

## 3. MUD Lifecycle Description for Network Operators

The totality of what network operators must do to build, operate, and maintain networks will not be described in exhaustive detail in this document. Instead, we will describe what additional or different things are necessary or recommended when establishing MUD support within the network. Some of the steps discussed will presuppose that networking equipment vendors will have added MUD support to their products.

The following high-level tasks are required to support the automatic network configuration aspects of MUD devices on the network:

1. Install and/or enable a MUD Policy Server

2. Configure network devices so that they will receive and enforce ACLs generated by the MUD Policy Server

3. Test and verify functionality by confirming that MUD files are retrieved and ACLs are applied to the appropriate ports and that those ACLs are removed when the port goes down

The MUD Policy Server may support caching retrieved MUD files. If it does, then the operator may choose to enable, tune, test, and monitor this functionality as well. Details about caching MUD files as well as each task above will be covered later in this document.

The network equipment to which MPDs connect must be capable of accepting and enabling dynamic ACLs which can preferrably be scoped to a port. While it is conceivable that the ACLs be combined and applied at a point in network that is multiple hops away from the switch to which the MPD connects, the tightest security controls are possible when enforcement can happen directly on the port. This eliminates the possibility that a MPD can talk to other devices on the same switch unless explicitly permitted. The remainder of this document will only discuss the case of using ACLs.

### 3.1. Installing and/or Enabling a MUD Controller

MUD Policy Servers can conceivably take on many forms, including stand-alone appliances, software modules installed on a switch or a router, a software package installed and integrated with a DHCP server, etc. The key requirements for MUD Policy Servers are:

1. Able to "see" a MUD URI

2. Able to retrieve a MUD file

For a MUD Policy Server to ‘‘see a MUD URI’’, it must either be able to see the DHCP or equivalent requests from MPDs directly or it must be otherwise connected to the service which does get to see these types of requests. For example the MUD Policy Server could be implemented as a plugin to a RADIUS server which is receiving requests from a switch which is handling DHCP requests by generating corresponding RADIUS AAA requests.

For a MUD Policy Server to be able to retrieve a MUD file, it must have network access permissive enough to retrieve files which are served from arbitrary locations on the internet.

Finally, to have any useful effect, the MUD Policy Server must be able to, having parsed a MUD file, generate ACLs which are to be applied to the appropriate port of the appropriate network device (i.e., a dynamic configuration must be generated and applied which reflects the MUD policy). The specifics of how the generated ACLs get back to the NAS and get applied to the proper port will depend on the design of the network.

At the time of this document’s preparation, MUD is still a new protocol and is under development. Therefore, descriptions of how it is integrated will be subject to adjustment according to the progression of actual implementations.

### 3.2. Network Device Configuration

There are two distinct "network configuration" concepts involved in the deployment of MUD:

1. Configuration of the network infrastructure such that the MUD controller is "in the loop" and able to issue configurations for devices as they appear on the network

2. The per-device dynamic configuration that is generated through the behavior of MUD itself

This document discusses both concepts where applicable. To avoid confusion, when a reference is made to "configuring a device" or similar, we will be referring to setting up the network infrastructure to include the MUD Policy Server into operations. The actions of the MUD infrastructure and network infrastructure to effect changes to network configurations persuant to MUD-advised policies will be referred to as "applying device policy" or (when it is more clear to do so) "applying the dynamic device configuration". The key word in the latter is dynamic and may be used when describing the specific steps being taken by the devices to apply the policies.

As previously mentioned, the ideal point for the application of MUD-based access restrictions is the port into which a device is directly plugged since this results in the most finely-grained application of access control and insures that devices are not able to talk even to neighbors on the same shared media without MUD authorization. For this to happen, the switches which connect to MUD-enabled devices must be configured to allow ACLs to be applied to each port. If the switch is stand-alone, then it will have to be configured to allow something like RADIUS or similar so that a controller device can send ACLs to the switch via an authorization transaction once the MUD profile has been processed.

For MUD to work properly, the switches MUST remove any dynamic configuration applied to a port when the connection on that port is dropped (such as when the cable to the port is disconnected). Once reconnected, a device will again issue a DHCP or similar request and the MUD behavior will begin again.

As an example, if a Layer-2 switch is used which can process DHCP requests by issuing RADIUS AAA requests to complete the port-level authorization, MUD process can occur by:

1. The switch adds the MUD URI to the RADIUS request (see [WEIS2017])

2. The RADIUS server passes the MUD URI to a MUD Controller

3. The returned MUD file is processed and the appropriate ACLs generated

4. The ACLs are encoded into the RADIUS Authorization response and returned to the switch

5. The switch receives the RADIUS Authorization, matches it to the port being provisioned, and applies the ACLs

### 3.3. Testing and Verification

In addition to the normal activities of validating through monitoring commands that ACLs have been applied as expected, the following items are suggested:

- o If one wants to understand what ACLs will be applied during a test of a particular device, one can read the MUD file to understand what access requirements it has and thus compare that with what ACLs get applied during the operation of the MUD protocol

- o The devices with MPDs attached to them should be checked to confirm the application of the expected ACLs and they are scoped to the appropriate ports

- o An ideal test would be to connect a MUD-enabled test client which will issue an appropriate network access negotiation via DHCP or whatever is appropriate for the NAS in use so that a full MUD File retrieval is triggered. The test client should then be used to try to both confirm connectivity to its explicity provisioned destination(s) while also verifying that it is not possible to reach sites outside the stipulated ACLs.

- o The MPD should be disconnected from the switch and the switch checked to verify that the ACLs are removed (which may not occur until another device is plugged into the same port)

### 3.4. Caching MUD Files

MUD Files may be cached by the MUD Controller. The MUD File itself indicates the minimum time between re-retrievals of a MUD File via the ‘‘cache-validity’’ attribute. When the MUD Controller is asked for a MUD File, if the URIs match a cached MUD File which is recent enough to be used, then that cached MUD File should be used. If not, then a valid MUD File MUST be retrieved by using the URI as a URL.

Note, however, that MUD files are very small. Additionally, MPDs will likely be installed into networks and then left running for long periods of time such that the number of MUD file requests will likely be small. Given those considerations, the value in caching MUD files, at least in the near term, is expected to be low.

## 4. Security Considerations

The bulk of this document describes the use of MUD to increase the security of a network. However, it is possible to compromise the effectiveness of MUD by attacking its behavior directly. This section discusses the known attacks and describes possible mitigations (all from the network operator's perspective). This section also attempts to clarify the limits to which MUD is expected to perform in terms of increasing security.

The use of MUD is intended to increase the level of security in the network relative to its current state. If the network has no security protections in place, then MUD may improve the situation by limiting access to MUD-enabled devices, but the network may already be too permissively accessible to be secure. A common comment about MUD is that a compromised MUD File can allow a MUD-enabled device to access arbitrary parts of the network or to allow arbitrary access to the device. If the network had had no security to begin with, then the compromised MUD File will not have reduce the security in any meaningful way.

To put this another way, any network SHOULD be properly designed such that the minimum required access is granted to all parties involved. If this is done, then a bad MUD File can only result in too permissive access to and from a single device in the network.

Although MUD is still a new protocol, it is conceivable that an "ecosystem" around it will grow that will enable a level of security validation that is much more difficult without it. In particular, the published MUD Files could be analyzed by third parties to assess their contents and to make users aware of anomalies. Additionally, deviations in successive versions of MUD Files can be audited to detect surprising changes.

Another commonly-mentioned attack scenario is tampering with the MUD URI during device bring-up to cause a different MUD File to be fetched and applied in place of the correct, manufacturer-supplied file. The ramifications of such an attack are no different than that of a compromised MUD File. The mitigation against the attack is insure the use of secure means of receiving and processing the device's advertisement of the MUD URI.

One other intriguing attack scenario is the spurious introduction of something akin to a "phantom" DHCP request with a MUD URI intended to coax the network infrastructure into fetching and acting on a MUD File, possibly without an actual device being present (or the "device" actually being a rogue software element running on a real device). In addition to mitigations already mentioned, port-level security should be used whenever possible with strict security policies to enable the detection of these rogue DHCP or other advertisements.

## 5. IANA Considerations

This document has no actions for IANA.

## 6. Normative References

[LEAR2017]
Lear, E., "Manufacturer Usage Description Specification", draft-ietf-opsawg-mud-03, January 05, 2017

[WEIS2017]
Weis, B., "RADIUS Extensions for Manufacturer Usage Description", draft-weis-radext-mud-00, October 25, 2016

## 7. Informative References

[RFC2882]
Mitton, D., "Network Access Servers Requirements: Extended RADIUS Practices", RFC2882, July 2000

## Authors’ Addresses

Steven Rich
Cisco Systems, Inc.
170 West Tasman Dr.
San Jose, CA 95134

Email: srich@cisco.com

Thorsten Dahm
Google Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043

Email: thorstendlux@google.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: September 9, 2017

J. White
D. Black
Dell EMC
J. Leung
Intel Corporation

March 9, 2017

# YANG Data Center Baseline Switch Profile
draft-wbl-rtgwg-baseline-switch-model-01

## Abstract

[ Insert abstract here ]

## Status of This Memo



## Copyright Notice

# 1. Introduction

*Disclaimer* - this is a -00 draft.

This is a normative profile for Baseline Switch Profile (send into IETF RTG) intended to be published as RFC on completion of DMTF spec to wrap Baseline Switch Profile.

# 2. What is a Redfish Baseline Switch?

The baseline switch profile contains basic system, interface, L2, and L3 configuration elements sufficient to set up the device for use in a controller based converged infrastructure environment.

The following list of IETF drafts, RFCs, and Redfish models will constitute the management interface to the baseline switch.

# 3. Core YANG RFCs

RFC6020 [1] provides the YANG modeling language definition.

RFC6991 [2] provides the Common YANG Data Types used by many other IETF YANG modules.

Interface management requires at set of RFCs to provide all relevant capabilities:

https://tools.ietf.org/html/rfc7223
https://tools.ietf.org/html/rfc7277
https://tools.ietf.org/html/rfc7224
https://tools.ietf.org/html/rfc7317

## 3.1. RFC7223 provides:

```
+--rw interfaces
|  +--rw interface* [name]
|     +--rw name                        string
|     +--rw description?                string
|     +--rw type                        identityref
|     +--rw enabled?                    boolean
|     +--rw link-up-down-trap-enable?   enumeration
+--ro interfaces-state
   +--ro interface* [name]
      +--ro name               string
      +--ro type               identityref
      +--ro admin-status       enumeration
      +--ro oper-status        enumeration
      +--ro last-change?       YANG:date-and-time
      +--ro if-index           int32
      +--ro phys-address?      YANG:phys-address
      +--ro higher-layer-if*   interface-state-ref
      +--ro lower-layer-if*    interface-state-ref
      +--ro speed?             YANG:gauge64
      +--ro statistics
         +--ro discontinuity-time    YANG:date-and-time
         +--ro in-octets?            YANG:counter64
         +--ro in-unicast-pkts?      YANG:counter64
         +--ro in-broadcast-pkts?    YANG:counter64
         +--ro in-multicast-pkts?    YANG:counter64
         +--ro in-discards?          YANG:counter32
         +--ro in-errors?            YANG:counter32
         +--ro in-unknown-protos?    YANG:counter32
         +--ro out-octets?           YANG:counter64
         +--ro out-unicast-pkts?     YANG:counter64
         +--ro out-broadcast-pkts?   YANG:counter64
         +--ro out-multicast-pkts?   YANG:counter64
         +--ro out-discards?         YANG:counter32
         +--ro out-errors?           YANG:counter32
```

### 3.2.  RFC7277 adds:

```
+--rw if:interfaces
  +--rw if:interface* [name]
     ...
     +--rw ipv4!
     |  +--rw enabled?            boolean
     |  +--rw forwarding?         boolean
     |  +--rw mtu?                uint16
     |  +--rw address* [ip]
     |  |  +--rw ip               inet:ipv4-address-no-zone
     |  |  +--rw (subnet)
     |  |     +--:(prefix-length)
```

```
          |  |     |  +--rw ip:prefix-length?   uint8
          |  |     +--:(netmask)
          |  |        +--rw ip:netmask?         YANG:dotted-quad
          |  +--rw neighbor* [ip]
          |     +--rw ip                     inet:ipv4-address-no-zone
          |     +--rw link-layer-address    YANG:phys-address
          +--rw ipv6!
             +--rw enabled?            boolean
             +--rw forwarding?         boolean
             +--rw mtu?                uint32
             +--rw address* [ip]
             |  +--rw ip               inet:ipv6-address-no-zone
             |  +--rw prefix-length    uint8
             +--rw neighbor* [ip]
             |  +--rw ip                     inet:ipv6-address-no-zone
             |  +--rw link-layer-address    YANG:phys-address
             +--rw dup-addr-detect-transmits?   uint32
             +--rw autoconf
                +--rw create-global-addresses?        boolean
                +--rw create-temporary-addresses?     boolean
                +--rw temporary-valid-lifetime?       uint32
                +--rw temporary-preferred-lifetime?   uint32
```

AND

```
    +--ro if:interfaces-state
       +--ro if:interface* [name]
          ...
          +--ro ipv4!
          |  +--ro forwarding?   boolean
          |  +--ro mtu?          uint16
          |  +--ro address* [ip]
          |  |  +--ro ip               inet:ipv4-address-no-zone
          |  |  +--ro (subnet)?
          |  |  |  +--:(prefix-length)
          |  |  |  |  +--ro prefix-length?   uint8
          |  |  |  +--:(netmask)
          |  |  |     +--ro netmask?         YANG:dotted-quad
          |  |  +--ro origin?          ip-address-origin
          |  +--ro neighbor* [ip]
          |     +--ro ip                     inet:ipv4-address-no-zone
          |     +--ro link-layer-address?   YANG:phys-address
          |     +--ro origin?               neighbor-origin
          +--ro ipv6!
             +--ro forwarding?   boolean
             +--ro mtu?          uint32
             +--ro address* [ip]
             |  +--ro ip               inet:ipv6-address-no-zone
```

```
             |  +--ro prefix-length    uint8
             |  +--ro origin?          ip-address-origin
             |  +--ro status?          enumeration
             +--ro neighbor* [ip]
                +--ro ip                    inet:ipv6-address-no-zone
                +--ro link-layer-address?   YANG:phys-address
                +--ro origin?               neighbor-origin
                +--ro is-router?            empty
                +--ro state?                enumeration
```

3.3.  RFC7224 provides:

The set of YANG identity statement for the IANA defined interface types.

3.4.  RFC7317 provides:

- o  System Identification

- o  System Time Date

- o  NTP

- o  DNS Client

System Identification

```
     +--rw system
     |  +--rw contact?          string
     |  +--rw hostname?         inet:domain-name
     |  +--rw location?         string
     +--ro system-state
        +--ro platform
           +--ro os-name?       string
           +--ro os-release?    string
           +--ro os-version?    string
           +--ro machine?       string
```

System Time

```
     +--rw system
     |  +--rw clock
     |  |  +--rw (timezone)?
     |  |     +--:(timezone-name)
     |  |     |  +--rw timezone-name?     timezone-name
     |  |     +--:(timezone-utc-offset)
     |  |        +--rw timezone-utc-offset?   int16
     |  +--rw ntp!
     |     +--rw enabled?   boolean
     |     +--rw server* [name]
     |        +--rw name                string
     |        +--rw (transport)
     |        |  +--:(udp)
     |        |     +--rw udp
     |        |        +--rw address    inet:host
     |        |        +--rw port?      inet:port-number
     |        +--rw association-type?   enumeration
     |        +--rw iburst?             boolean
     |        +--rw prefer?             boolean
     +--ro system-state
        +--ro clock
           +--ro current-datetime?      YANG:date-and-time
           +--ro boot-datetime?         YANG:date-and-time
```

DNS Client

```
     +--rw system
        +--rw dns-resolver
           +--rw search*    inet:domain-name
           +--rw server* [name]
           |  +--rw name    string
           |  +--rw (transport)
           |     +--:(udp-and-tcp)
           |        +--udp-and-tcp
           |           +--rw address    inet:ip-address
           |           +--rw port?      inet:port-number
           +--rw options
              +--rw timeout?    uint8
              +--rw attempts?   uint8
```

User Authentication

```
+--rw system
   +--rw authentication
      +--rw user-authentication-order*   identityref
      +--rw user* [name]
         +--rw name        string
         +--rw password?   ianach:crypt-hash
         +--rw authorized-key* [name]
            +--rw name         string
            +--rw algorithm    string
            +--rw key-data     binary
```

## 4. Additional YANG models

In addition to the above RFCs, the baseline switch models needs to cover:

- VLANs
- ACLs
- Syslog

The following lists of IETF drafts sets our recommendation to cover the above three areas.

### 4.1. VLAN and interface extensions:

To handle VLANs and with related interface configuration the following YANG models are under evaluation.

- https://tools.ietf.org/html/draft-ietf-netmod-intf-ext-yang-03
- https://tools.ietf.org/html/draft-wilton-intf-vlan-yang-00.txt ## ACL To handle ACL configuration the following YANG model is under consideration.
- https://tools.ietf.org/html/draft-ietf-netmod-acl-model-09

### 4.2. Syslog

To handle configuration and access to syslog the following YANG model is under consideration.

- https://tools.ietf.org/html/draft-ietf-netmod-syslog-model-11

## 5. Applicable Redfish system management models

The following standard Redfish systems management models apply to the baseline network switch profile. Reference: Redfish schema index [3]. The use of these Redfish management models allows a converged infrastructure manager to have a consistent view of server, storage and network systems.

- Chassis
- ComputerSystem
- Manager
- ManagerAccount
- Power
- Thermal
- SoftwareInventory plus UpdateService
- Event configuration using Event, EventDestination, and Event Service
- Access to logs using LogEntry, and LogService
- Management interface configuration using EthernetInterface and related
- Console configuration using SerialInterface
- PrivilegeRegistery and Privileges

Where YANG and Redfish overlap, the commonality of YANG vs Redfish is TBD.

## 6. Overall Baseline Switch Profile Structure

```
./redfish/v1/Systems
./redfish/v1/Chassis
./redfish/v1/NetworkDevices/BaselineSwitch/...
... other redfish resource blocks...
(resource from RFCs and Redfish bullet list, above)
```

# 7. References

## 7.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

## 7.2. URIs

[1] https://tools.ietf.org/html/rfc6020

[2] https://tools.ietf.org/html/rfc6991

[3] http://redfish.dmtf.org/redfish/schema_index

# Authors' Addresses

Joseph White
Dell EMC

Email: joseph.l.white@dell.com

David Black
Dell EMC

Email: david.black@dell.com

John Leung
Intel Corporation

Email: john.leung@intel.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: September 9, 2017
J. White
D. Black
Dell EMC
J. Leung
Intel Corporation
March 9, 2017

# YANG Baseline Switch Profile Background
draft-wbl-rtgwg-yang-ci-profile-bkgd-01

## Abstract

A YANG device profile is primarily a group of YANG models that are appropriate for use with a particular class of device or in specific device roles. This document provides background and describes the rationale for a baseline data center switch device profile, e.g., for top-of-rack switches in data center converged infrastructure. This rationale is based on the reuse of YANG models by the DMTF Redfish standard for management of converged infrastructure, but the resulting YANG device profile is intended to be useable by any YANG-based network management approach or protocol, as opposed to being specific to Redfish.

## Status of This Memo



## Copyright Notice

## 1. Introduction

*Disclaimer* - this is a -00 draft, whose primary goal is to capture the rationale for use of YANG for Redfish network management and the desirability of a baseline data center network switch profile, including providing technical background on the Redfish standard and its approach to network management. The draft content is not polished, and the organization of the material is likely to change.

A YANG device profile is primarily a group of YANG models that are appropriate for use with a particular class of device or in specific device roles. This document provides background and describes the rationale for a baseline data center switch device profile, e.g., for top-of-rack switches in data center converged infrastructure. The rationale is based on the reuse of YANG models by the DMTF Redfish standard for management of converged infrastructure, but the resulting YANG device profile is intended to be useable by any YANG-based network management approach or protocol; it is not intended to be Redfish-specific.

In support of this rationale, this document provides background on how YANG is used in the Redfish management framework; the YANG modules are translated to Redfish schema definitions in order to enable reuse of the models are with Redfish-defined management protocols and related functionality.

## 2. Motivation

A common management framework with accompanying set of protocols and device models is desirable in systems management use cases. A good example of this is in a converged infrastructure deployment within a data center. Applications, servers, storage, appliances, and networking elements are assembled to create a combined coherent platform. For the networking components in such an environment, there are platform management elements that are common with other types of systems such as thermal monitoring, physical enclosure, fans, and power supplies, as well as networking specific management elements to control the forwarding and filtering of network packets or the networking services. The common elements should be accessed and managed in a single way across all systems within the deployment.

Management, orchestration, and control of such a system benefits from a single approach that enables unified tools sets and simplifies operations.

The networking specific configuration within the converged infrastructure environment only needs a subset of all the possible networking protocols and services giving the converged controller only the minimum spanning control surface in terms of the models it can access. A breakdown of the needs of such a switch result in about 5-10 YANG modules (see Appendix A). These 5-10 modules should lead to common YANG-based data center network switch management across vendors and products.

As a contrast, a full function edge router would need many more protocols and services along with full function virtualization resulting in the use of about 80 YANG modules.

### 2.1. Redfish Background

The DMTF (Distributed Management Task Force) Redfish standard is becoming the common standard for converged infrastructure (CI) management. Converged Infrastructure consists of rack-scale (partial or full-rack) integrated compute, network and storage infrastructure that is procured and deployed as rack scale systems.

Redfish data center storage management functionality has been extended in partnership with SNIA (Storage Networking Industry Association) resulting in the recently released Swordfish specification that extends Redfish for networked storage and storage network management. The authors hope to work with the IETF to extend and align Redfish network management with IETF’s YANG framework.

Redfish is a management standard using a data model representation inside of a RESTful interface. The model is expressed in terms of a standard, machine-readable schema, with the payload of the messages being expressed in JSON.

Because it is a model-based API, Redfish is capable of representing a variety of implementations via a consistent interface. It has mechanisms for managing data center resources, handling events, long lived tasks and discovery, as well.

In Redfish, every URL represents a resource. This could be a service, a collection, an entity or some other construct. But in RESTful terms, these URIs point to resources and Redfish clients interact with these resources. For example, the content of a resource, in JSON format, is returned when the Redfish client performs a HTTP GET.

OData is an OASIS standard for expressing the schema of a JSON document. OData allows a fuller description of the JSON document, than json-schema. The format of OData schema is specified in CSDL (Common Schema Definition Language).

OData also describes directives that can appear on the URI to control the contents of the HTTP response. In Redfish, these directives (i.e. $top and $skip) are stated as 'should'. Networking extension may change the requirement to 'shall'.

Redfish releases include both OData and json-schema schema. With json-schema, users can take advantage of its larger tool chain.

Additional information about OData can be found at the OData site [1].

Additional information about Redfish can be found at DMTF's Redfish site [2]. Specifically, the Redfish White Paper [3] provides a good overview.

## 2.2. YANG and Redfish

Within the networking world, YANG has become the preferred management framework. YANG expresses each section of the overall model as a module containing a tree of nodes where each node is either a container node that builds the hierarchy or a leaf node containing data elements for the model. Redfish network management leverages YANG as the core model definition language to maintain consistency with other YANG based network management approaches. Using a common model structure results in equivalent data elements yielding the same data or content when accessed via different interfaces or protocols.

Redfish's network management supports this consistency by reusing YANG modules as Redfish models for network functionality and services, mechanically translating those modules to the Redfish interface, based on HTTP, JSON, and OData.

The Redfish approach to network management enables definitions of a specific system views or profiles that are aligned with the configuration functionality required in a specific scenario or for a specific class of network devices. A particularly relevant example is a baseline switch model description with a minimum set of model elements; this is useful when configuring a switch within a larger converged infrastructure system. The model elements of the baseline switch should be the smallest set of models necessary to configure a data center switch in a converged infrastructure environment; the corresponding set of YANG modules would be a simple data center network device profile. A more complex network router might need tunnel models, overlay models, extensive QoS models, and other elements.

The top level resource structure of Redfish is show below.

```
./redfish/v1/Systems
./redfish/v1/Chassis
./redfish/v1/NetworkDevices
(other redfish resources)
```

Within this structure a network switch is viewed as a data center element for its common subsystems such as chassis, power, thermal, cooling, management access setup, etc while the network modeling is specified within the instances of the NetworkDevices[] collection. Each member of the NetworkDevices[] collection has implements its own set translated YANG modules. For consistency, the set of modules grouped under a NetworkDevice instance can follow one of multiple standard groupings expressed as a profile to manage different classes of equipment and satisfy different use cases. Two profile examples are the basic switch and virtualized edge router.

## 2.3. YANG mapping to Redfish

The notion of schema is different in Redfish and YANG.

In YANG, a schema is device specific. The user determines the YANG modules utilized by a system, and may consult a module library as part of doing so (e.g., RFC7895 [4]). The YANG schema is realized as a set of YANG modules, each with a prescribed node tree structure.

In Redfish, there is one schema that encompasses the entire namespace. In other words, Redfish has a global namespace for its schema, of which the device implements a subset. The Redfish schema is realized as resources accessed via a URI, so the namespace contains the information about which YANG modules are utilized. The OData directives $expand and $filter allows the client to discovery this directly from the parent namespace node above the modules.

That functionality obviates any Redfish need to use YANG module combination techniques such as YANG Schema-mount [5].

Despite these differences, the proposed profiles should be usable by both YANG based protocols (e.g., NETCONF, RESTCONF) and Redfish, as the core content of each profile is a set of YANG modules.

To allow Redfish to manage network devices, the YANG modules needs to be translated into OData CSDL (Common Schema Definition Language). The translation is specified in the YANG-to-Redfish Mapping

Specification [6]. The translation has the following characteristics:

- o Includes, imports, and augments, are compiled out to create a single consistent schema block constituting a particular instance of a NetworkDevice.

- o The YANG node tree layout is reflected in the URI layout

- o The individual YANG container nodes and list nodes are rendered as resources with the YANG tree hierarchy reflected as navigation properties.

Access to the YANG data model elements uses a Redfish JSON accessed via a provider on the URI target.

Leaf nodes representing common back end system "features or elements" return consistent data independent of node name and network device hierarchy.

The NetworkDevices[] collection allows

- o Multiple co-existing and consistent views onto a system.

  * Horizontally extensible

  * Vertical hierarchy allowing for control interface delegation

- o This is similar to a "view class" or facade approach in software.

### 2.4. Example Mapping

The following shows the resource which results from mapping RFC7223 (ietf_interface module) to the Redfish schema. Below is a fragment of the data model from the RFC.

```
+--rw interfaces
| +--rw interface* [name]
| +--rw name string
| +--rw description? string
| +--rw type identityref
| +--rw enabled? boolean
| +--rw link-up-down-trap-enable? enumeration
+--ro interfaces-state
+--ro interface* [name]
+--ro name string
+--ro type identityref
+--ro admin-status enumeration
```

The translation to Redfish CSDL is performed using the RFC’s YANG code. The translation will generate the CSDL files for the ietf_interfaces resource and each YANG container. The path to these resources mirror the above data model.

```
./redfish/v1/NetworkDevices/Switch1
./redfish/v1/NetworkDevices/Switch1/ietf_interfaces
./redfish/v1/NetworkDevices/Switch1/ietf_interfaces/interfaces
./redfish/v1/NetworkDevices/Switch1/ietf_interfaces/interfaces/ethernet1
./redfish/v1/NetworkDevices/Switch1/ietf_interfaces/interfaces_state
...
```

A HTTP GET of the "ethernet1" singleton resource will return the following JSON document. Note that each property from the above data model is present in the resource.


```
{
    "Id": "ethernet1",
    "Name": "ethernet1",
    "Description": "Ethernet interface on slot 1",
    "type": "iana_if_type:ethernetCsmacd",
    "enabled": "true",
    "link_up_down_trap_enable": "true"

    "@odata.context":
        "/redfish/v1/$metadata#ietf_interfaces.interfaces.interface.
            interface",
    "@odata.type": "#interface.v1_0_0.interfaces",
    "@odata.id":
        "/redfish/v1/NetworkDevices/Switch1/ietf_interfaces
            /interfaces/ethernet1"
}
```


The three properties at the end of the JSON document are OData annotations.

## 3. Security Considerations

Redfish also improves security control since there is a single point of management contact for a device to control all of its functions.

(Additional security discussion will be provided later.)

## 4. Appendix A

YANG models needed to managed a network switch:

o RFC7223 (Interfaces)

- o RFC7224 (IANA)
- o RFC7277 (IPv4, IPv6)
- o RFC7317 (System Identification, Time-Date, NTP)
- o VLANs
- o ACLs
- o Syslog

## 5. Appendix B

The following describes how the Redfish NetworkDevices[] collection resource allows multiple co-existing and consistent views onto a system.

As an example, a router could have the following:

```
//redfish.example.net/redfish/v1/NetworkDevices/masterRouter
//redfish.example.net/redfish/v1/NetworkDevices/vrf1
//redfish.example.net/redfish/v1/NetworkDevices/vrf2
```

In this example, masterRouter represents the complete system with all interfaces, all tables, all system level configuration, and a model structure for assigning resources to virtual instances. The resources, vrf1 and vrf2, represent a particular partitioning of the system to create virtual router instances each assigned a subset of the total resource pool.

The above structure has similarities with that expressed by the device model from the following references:

- o https://tools.ietf.org/html/draft-ietf-rtgwg-device-model-01
- o https://tools.ietf.org/html/draft-ietf-rtgwg-ni-model-01
- o https://tools.ietf.org/html/draft-ietf-rtgwg-lne-model-01
- o https://tools.ietf.org/html/draft-ietf-netmod-schema-mount-03

In these references a Network Device contains Logical Network Elements which, in turn, contain Network Instances. From the device model reference, the Network Device represents the system as a whole. The Logical Network Element represents a partition of a physical system. The Logical Network Element represents a VRF or VSI (virtual switching instance).

The Redfish NetworkDevices collection resource would map this modeling approach by using an element of the collection for the Network Device and one for each of the Logical Network Elements and Network Instances. These collection elements would add references at the NetworkDevices element level to map the containment of of the device model. The overall ./redfish/v1/ root maps to the Routing Area Network Device.

## 6. Appendix C

The following is the ietf_interfaces.interfaces.interface_v1.xml CSDL metadata file, which is referenced in @odata.context annotation in the example mapping. The entity type referenced in the @odata.type annotation is in the second Namespace.

When mapping YANG code to CDSL, values are mapped to existing OData core properties, when possible. Otherwise, new annotations are defined in RedfishYangExtensions.xml. This file is referenced at the beginning of the document.

```
<?xml version="1.0" encoding="UTF-8"?>
<edmx:Edmx xmlns:edmx="http://docs.oasis-open.org/odata/ns/edmx"
        Version="4.0">
    <Edmx:Reference
        Uri="http://docs.oasis-open.org/odata/odata/v4.0/cs01/
            vocabularies/Org.OData.Core.V1.xml">
        <Edmx:Include Alias="Odata" Namespace="Org.OData.Core.V1"/>
    </Edmx:Reference>
    <Edmx:Reference
        Uri="http://docs.oasis-open.org/odata/odata/v4.0/cs01/
            vocabularies/Org.OData.Capabilities.V1.xml">
        <Edmx:Include Alias="Odata"
            Namespace="Org.OData.Capabilities.V1"/>
    </Edmx:Reference>
    <Edmx:Reference
        Uri="http://redfish.dmtf.org/schemas/v1/
            RedfishYangExtensions.xml">
        <Edmx:Include Alias="Redfish.Yang"
            Namespace="Redfish.Yang"/>
    </Edmx:Reference>

    <Edmx:DataServices>

    <Schema Namespace="interface"
            xmlns="http://docs.oasis-open.org/odata/ns/edm" >
        <EntityType Name="interface"
                BaseType="Resource.v1_0_0.Resource">
            <Annotation Term="OData.Description"
```

```
                String="<manual input>." />
            <Annotation Term="OData.AdditionalProperties"
                Bool="False"/>
        </EntityType>
    </Schema>

    <Schema Namespace="interface.v1_0_0"
            xmlns="http://docs.oasis-open.org/odata/ns/edm" >
        <EntityType Name="interface" BaseType=
                "ietf_interfaces.interfaces.interface.interface" >
            <Annotation Term="OData.Description"
                String="<manual input>." />
            <Annotation Term="OData.AdditionalProperties"
                Bool="False"/>
            <Annotation Term="Redfish.Yang.NodeType"
                EnumMember="Redfish.Yang.NodeTypes/list" />
            <Annotation Term="Redfish.Yang.key"
                String=" the yang key string"/>
            <Key>
                <PropertyRef Name="name"/>
            </Key>
            <Property Name="name" Type="Edm:String">
                <Annotation Term="OData.Description"
                    String="..." />
                <Annotation Term="OData.Permissions"
                    EnumMember="OData.Permissions/Read"/>
                <Annotation Term="Redfish.Yang.NodeType"
                    EnumMember="Redfish.Yang.NodeTypes/leaf"/>
                <Annotation Term="Redfish.Yang.YangType"
                    String="string" />
            </Property>
            <Property Name="description" Type="Edm:String">
                <Annotation Term="OData.Description"
                    String="..." />
                <Annotation Term="OData.Permissions"
                    EnumMember="OData.Permissions/Read"/>
                <Annotation Term="Redfish.Yang.NodeType"
                    EnumMember="Redfish.Yang.NodeTypes/leaf" />
                <Annotation Term="Redfish.Yang.YangType"
                    String="string" />
                <Annotation Term="Redfish.Yang.reference"
                    String="RFC 2863: The Interfaces Group..." />
            </Property>
            <Property Name="type" Type="Edm:String">
                <Annotation Term="OData.Description" String="..."/>
                <Annotation Term="Redfish.Yang.NodeType"
                    EnumMember="Redfish.Yang.NodeTypes/leaf" />
                <Annotation Term="Redfish.Yang.YangType"
```

```
                        String="identityref"/>
                    <Annotation Term="Redfish.Yang.base"
                        String="interface-type"/>
                    <Annotation Term="Redfish.Yang.mandatory"
                        EnumMember="Redfish.Yang.Mandatory/true"/>
                    <Annotation Term="Redfish.Yang.reference"
                        String="RFC 2863: The Interfaces Group..." />
                </Property>
                <Property DefaultValue="true" Name="enabled"
                        Type="Edm:Boolean">
                    <Annotation Term="OData.Description"
                        String="This leaf contains..." />
                    <Annotation Term="Redfish.Yang.NodeType"
                        EnumMember="Redfish.Yang.NodeTypes/leaf" />
                    <Annotation Term="Redfish.Yang.YangType"
                        String="boolean"/>
                    <Annotation Term="Redfish.Yang.reference"
                        String="RFC 2863: The Interfaces..."/>
                </Property>
                <Property Name="link_up_down_trap_enable"
                        Type="Edm:Enumeration">
                    <Annotation Term="OData.Description"
                        String="Controls whether..."/>
                    <Annotation Term="Redfish.Yang.NodeType"
                        EnumMember="Redfish.Yang.NodeTypes/leaf"/>
                    <Annotation Term="Redfish.Yang.YangType"
                        String="enumeration"/>
                    <Annotation Term="Redfish.Yang.if_feature"
                        String="if-mib"/>
                    <Annotation Term="Redfish.Yang.reference"
                        String="RFC 2863: The Interfaces..." />
                    <EnumType
                            Name="link_up_down_trap_enableEnumeration">
                        <Member Name="enabled" Value="1">
                            <Annotation Term="Redfish.Yang.enum"
                                String="enabled"/>
                        </Member>
                        <Member Name="disabled" Value="2">
                            <Annotation Term="Redfish.Yang.enum"
                                String="disabled"/>
                        </Member>
                    </EnumType>
                </Property>
            </EntityType>
        </Schema>

        </Edmx:DataServices>
```

```
</edmx:Edmx>
```

# 7. Appendix D

The following is the IETF YANG source XML from RFC7223 used for the example mapping.

```
<CODE BEGINS> file "ietf-interfaces@2014-05-08.yang"
module ietf-interfaces {
    namespace "urn:ietf:params:xml:ns:yang:ietf-interfaces";
    prefix if;
    import ietf-yang-types {
        prefix yang;
    }
    organization
        "IETF NETMOD (NETCONF Data Modeling Language) Working Group";
    . . .
```

After the typedef, identity, and feature statements, the data model is defined.  Below is the fragment that becomes ietf_interfaces.interfaces.interface_v1.xml.

```
    /*
     * Configuration data nodes
     */
    container interfaces {
        description
            "Interface configuration parameters.";
        list interface {
            key "name";
            description
                "The list of configured interfaces...";
        leaf name {
            type string;
            description
                "The name of the interface...";
        }
        leaf description {
            type string;
            description
                "A textual description of the interface...";
            reference
                "RFC 2863: The Interfaces Group MIB - ifAlias";
        }
        leaf type {
            type identityref {
                base interface-type;
            }
```

```
                mandatory true;
                description
                    "The type of the interface...";
                reference
                    "RFC 2863: The Interfaces Group MIB - ifType";
            }
            leaf enabled {
                type boolean;
                default "true";
                description
                    "This leaf contains the configured,...";
                reference
                    "RFC 2863: The Interfaces Group MIB - ifAdminStatus";

            leaf link-up-down-trap-enable {
                if-feature if-mib;
                type enumeration {
                    enum enabled {
                        value 1;
                    }
                    enum disabled {
                        value 2;
                    }
                }
                description
                    "Controls whether linkUp/linkDown SNMP...";
                reference
                    "RFC 2863: The Interfaces Group MIB -
                        ifLinkUpDownTrapEnable";
            }
        }
        . . .
```

# 8. References

## 8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

## 8.2. URIs

[1] http://odata.org

[2] http://dmtf.org/redfish

[3] http://www.dmtf.org/sites/default/files/standards/documents/DSP2044_1.0.0.pdf

[4] http://www.rfc-editor.org/info/rfc7895

[5] https://tools.ietf.org/html/draft-ietf-netmod-schema-mount-03

[6] http://www.dmtf.org/sites/default/files/standards/documents/DSP0271_0.5.6.pdf

Authors' Addresses

Joseph White
Dell EMC

Email: joseph.l.white@dell.com

David Black
Dell EMC

Email: david.black@dell.com

John Leung
Intel Corporation

Email: john.leung@intel.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: September 14, 2017

Y. Zheng
China Unicom
S. Xu
D. Dhody
Huawei Technologies

March 13, 2017

# Usecases for Network Artificial Intelligence (NAI)
draft-zheng-opsawg-network-ai-usecases-00

## Abstract

This document discusses the scope of Network Artificial Intelligence (NAI), and the possible use cases that are able to demonstrate the advantage of applying NAI.

## Status of This Memo



## Copyright Notice

Table of Contents



## 1. Introduction

Current networks have become much more dynamic and complex, and pose new challenges for network management and optimization. For example, network management/optimization should be automated to avoid human intervention (and thus to minimize the operational expense). Artificial Intelligence (AI) and Machine Learning (ML) is a promising approach to realize such automation, and can even do better than human beings. Furthermore, the population of Software-Defined Networks (SDN) paradigm makes the application of Artificial Intelligence in networks possible, since the SDN controller has the complete knowledge of the network status and can control behavior of network nodes to implement AI decisions.

AI and ML technologies can learn from historical data, and make predictions or decisions, rather than following strictly static program instructions. They can dynamically adapt to a changing situation and enhance their own intelligence with by learning from new data. It can learn and complete complicated tasks. It also has potential in the network technology area especially with SDN and Network Function Virtualization (NFV).

This document presents the concept of Network Artificial Intelligence. It first discusses the scope of Network Artificial Intelligence (NAI). And then Some use cases are discussed to demonstrate the advantage of applying NAI.

## 2. NAI Architecture

The definition of the architecture of NAI could be refer to [I-D.li-rtgwg-network-ai-arch]. In the architecture of NAI, central controller is the core part of Network Artificial Intelligence which can be called as 'Network Brain'. The Network Telemetry and Analytics (NTA) engines can be introduced acompanying with the central controller. The Network Telemetry and Analytics (NTA) engine inclues data collector, analytics framework, data persistence, and NAI applications.

```
                ^                                   ^
             (4)|                                   |(4)
+---------------|--------------+    +---------------|--------------+
| Domain 1      |              |    |               |     Domain 2 |
|        +------------+        |    |        +------------+        |
|        |  Central   |        |    |        |  Central   |        |
|     (1)| Controller |----------------------| Controller |(1)     |
|        |    with    |        |    |        |    with    |        |
|        |     NTA    |        |    |        |     NTA    |        |
|        +------------+        |    |        +------------+        |
|         /          \         |    |         /          \         |
|     (3)/            \        |    |        /            \(3)     |
|       /              \       |    |       /              \       |
| +--------+        +--------+ |    | +--------+        +--------+ |
| |        |        |        | |    | |        |        |        | |
| |Network | ...... |Network | |    | |Network | ...... |Network | |
| | Device |  (2)   | Device | |    | | Device |  (2)   | Device | |
| |    1   |        |    N   | |    | |    1   |        |    N   | |
| +--------+        +--------+ |    | +--------+        +--------+ |
|                              |    |                              |
+------------------------------+    +------------------------------+
```

Figure 1: An Architecture of Network Artificial Intelligence(NAI)

## 3. NAI Use Cases

### 3.1. Traffic Predication and Re-Optimization/Adjustment

This subsection introduces the Path Computation Element (PCE) [RFC4655] use cases in wide area networks (WAN). In PCE scenario, network data collection is realized through the control plane protocols such as PCE protocol (PCEP) and BGP-LS [RFC7752] protocol and data are passed to the PCE application. PCEP receives the state of Label Switched Path (LSP) from the network, and BGP-LS receives the topology information from the network. If network telemetry is used, traffic information can be received from the network as well directly at the NTA engine using protocols such as gRPC.

PCE application (APP) only maintains the latest information. To enable NAI, history of all LSP and topology changes is stored in external data repository. Further traffic monitoring data could also be collected and stored, if network telemetry is used. There are two usecases in the application scenarios: (1) reroute/re-optimize using the historical trend and predications from AI; (2) traffic congestion avoidance and AI-enabled auto-bandwidth adjustment.

For the usecase (1), the analytics component in NTA (Network Telemetry and Analytics), can use stored data to build models to predict impact of network events and state of the LSPs. For example, it can use historical trends to guide path computation to include/exclude specific links. Finding correlations between data, finding anomalies and data visualization are also possible.

The analytics component in NTA can also use stored data to detect and predict network events and request PCE to take necessary actions. For example, it can use network bandwidth utilization historical trends to request for re-optimizations.

For the usecase (2), with network telemetry, the NTA can collect per-link and per-LSP traffic flow using gRPC from network. Such network telemetry data includes statistics for tunnels, links, bandwidth reservations, actual usage, delay, jitter, packet loss, etc. Meanwhile, it also collects data regarding network events and its impact on traffic flows. The analytics component can use telemetry data to build traffic models to predict traffic congestion when new years or sporting events are coming. According to the congestion prediction, the PCE app could reroute traffic to avoid congested links. Besides the case, NTA can also perform predication and make necessary changes to network. In particular, the PCE APP performs bandwidth usage prediction (i.e., bandwidth calendaring) by looking at the historical trends of all sampled data instead of the instant sampled data. The collected data are traffic engineering data base (TEDB) and LSP-DB, and can also include scheduling information. In addition, the collected data also include auto-bandwidth related changes under particular network events. Using machine learning algorithm, the analytics component is able to correct such changes with the events, and predicts network events and their impact.

## 3.2. Route Monitoring and Analytics

This subsection introduces the BGP Monitoring Protocol (BMP) [RFC7854] use case in wide area networks (WAN). The BGP protocol is known for its flexibility and ability to manage a large number of neighbors and routes. It is also the basis for many overlay services such as L3VPN, L2VPN and so on. The BMP protocol can be used by the controller to monitor BGP protocol neighbor status and routing information on the routers.

According to [RFC7854], BMP client located in the router collects BGP neighbor status, routes for each neighbor, and events defined by the user. And then it passes the informations through the BMP protocol to the management station located on the controller. Based on BMP monitoring of BGP, there are three use cases: (1) BGP Route Leaks Monitoring; (2) BGP Hijacks Monitoring; (3) Traffic Analytics.

Route leaks involve the illegitimate advertisement of prefixes, blocks of IP addresses, which propagate across networks and lead to incorrect or suboptimal routing. For case (1), based on BMP, NAI apps can analyze BGP route leaks.

For case (2), by manipulating BGP, data can be rerouted in an attacker's favor out them to intercept or modify traffic.If the malicious announcement is more specific than the legitimate one, or claims to offer a shorter path, the traffic may be directed to the attacker.By broadcasting false announcements, the compromised router may poison the RIB of its peers.After poisoning one peer, the malicious routing information could propagate to other peers, to other Autonomous Systems, and onto the interactive Internet. Based on monitoring BGP routes, ML algorithms can be trained to determine when a hijack has taken place and take necessary actions.

In case (3), with BMP protocol providing BGP changes, together with Telemetry providing network traffic information, The NAI Apps can analyze traffic trends, predict traffic changes, and do traffic optimizing.

### 3.3. Multilayer Fault Detection In NFV Framework

The high reliability and high availability required for carrier-class applications is a big challenge in virtualized and software-based environment where failures are normal in a software-based environment. The interdependence between NFV's abstraction levels and virtual resources is complex as shown in Fig.. The dynamic characteristics of the resources in the cloud environment make it difficult to locate the fault. So multilayer fault detection for NFV networks and cloud environment will be very useful.

```
                         +--------------------+
                         |      Central       |
                         |     Controller     |
                         |        with        |
                         |         NTA        |
                         +--------------------+
                             |      |      |
                             |      |      |
                             |      |      |
                             V      V      V
        +-------------------------------------------------------+
        |                                                       |
        |      +-----------+  +-----------+  +-----------+      |
        |      |    VNF1   |  |    VNF2   |  |    VNF3   |      |
        |      +-----|-----+  +-----|-----+  +-----|-----+      |
        |            |              | VN-NF        |            |
        |    +-------|--------------|--------------|-------+    |
        |    | NFVI                                        |    |
        |    | +-----------+  +-----------+  +-----------+ |    |
        |    | |  Virtual  |  |  Virtual  |  |  Virtual  | |    |
        |    | | Computing |  |  Storage  |  |  Network  | |    |
        |    | +-----------+  +-----------+  +-----------+ |    |
        |    | +-----------------------------------------+ |    |
        |    | |        VIRTUALIZATION LAYER             | |    |
        |    | +--------------------|--------------------+ |    |
        |    |                VI-Ha |                      |    |
        |    |+---------------------|---------------------+|    |
        |    ||                         Hardware Resouces ||    |
        |    ||+-----------+  +-----------+  +-----------+||    |
        |    ||| Computing |  |  Storage  |  |  Network  |||    |
        |    |||  Hardware |  |  Hardware |  |  Hardware |||    |
        |    ||+-----------+  +-----------+  +-----------+||    |
        |    |+-------------------------------------------+|    |
        |    +---------------------------------------------+    |
        |                                                       |
        +-------------------------------------------------------+
```

Figure 2 NAI in Multi-layer NFV Framework

For the virtualization layer, CPU performance, memory usage, interface bandwidth and other KPI indicators can be monitored. At the same time resource occupancy and the life cycle of NVF software process can also be monitored. Through the NAI, the relevant statistical data in multiple levels can be analyzed and the models can be setup to locate the root cause for the possible fault in the multi-layer environment.

### 3.4. Data Center Network Use Cases

Traditionally, data center networks have comprised a large number of switches and routers that direct traffic based on the limited view of each device. With help of SDN/NFV the data center networks are more agile and dynamic to changing usage and traffic patterns. The real-time traffic data and usage can be used to make the data center management and operations intelligent.

Various protocols such as sFLOW, IPFIX could be used to get the port statistics as well as traffic sampling. Over time this information can help build the traffic usage models on a per port and per flow basis. With historical data as the base the NTA engine can predict the traffic usage and make necessary instructions to the SDN controller or NFV orchestrator. These instructions could be reroute a flow to avoid a congested port or scale-in another switch to share load based on the predicted traffic demand.

The NTA engine should find correlation between the various network data to build models and predict the impact of network events, congestions, network utilization patters etc. Further NTA could detect anomalies based on the historical patterns and help in root cause analysis. The policy framework can be enhanced to consider the analytics.

NTA engine could also get the usage and health information from the Host (servers). Correlation between this information with the information received from network could help in finding security flows and anomalies when the information does not match.

#### 3.4.1. Service Function Chaining

This sub section introduces how to apply NAI to SFC scenario to intelligently reroute/re-optimize the service chains; increase utilization for both Service Functions(SF) and network; intelligent selection of the Service Function Path (SFP) based on data traffic trends.

As per [RFC7665], Service function chaining (SFC) enables the creation of composite (network), services that consist of an ordered set of SFs that must be applied for specific treatment of received packets and/or frames and/or flows selected as a result of classification The SFs of chain are connected using a service function forwarder (SFF), which is responsible for forwarding traffic to one or more connected SFs according to information carried in the SFC encapsulation, as well as handling traffic coming back from the SF.

The various network telemetry information like delay, jitter, packet loss from the network and the CPU/memory usage utilizations from the SFs, can be collected using sFLOW/gRPC protocol and stored in persistent data repository. The analytics component in NTA can use stored data to build statistics models to predict the impact on various Service Function Paths due to network events, traffic and state of the SFPs and instruct the SDN controller to take necessary actions SDN controller can calculate new paths/reroute the SFC path to avoid congested Ports/SFFs or overloaded SFs. This correlation of application analytics from the SFs and the network analytics from the SFFs could enhance the intelligent management of the service chains for the operators.

The usage and traffic pattern over time can help increase the utilization of SF as well as the underlay network.

## 4. Contributors

The following people have substantially contributed to the usecases of NAI:

Lizhao You
Huawei
Email: youlizhao@huawei.com

Kalyankumar Asangi
Huawei
Email: kalyana@huawei.com

## 5. Security Considerations

TBD

## 6. IANA Considerations

This document has no actions for IANA.

## 7. Acknowledgement

Thanks to Li Zhenbin and Liu Shucheng for their comments and contribution.

## 8. References

### 8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

### 8.2. Informative References

[I-D.li-rtgwg-network-ai-arch] Li, Z. and J. Zhang, "An Architecture of Network Artificial Intelligence(NAI)", draft-li-rtgwg-network-ai-arch-00 (work in progress), October 2016.

[RFC4655] Farrel, A., Vasseur, J., and J. Ash, "A Path Computation Element (PCE)-Based Architecture", RFC 4655, DOI 10.17487/RFC4655, August 2006, <http://www.rfc-editor.org/info/rfc4655>.

[RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <http://www.rfc-editor.org/info/rfc7665>.

[RFC7752] Gredler, H., Ed., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP", RFC 7752, DOI 10.17487/RFC7752, March 2016, <http://www.rfc-editor.org/info/rfc7752>.

[RFC7854] Scudder, J., Ed., Fernando, R., and S. Stuart, "BGP Monitoring Protocol (BMP)", RFC 7854, DOI 10.17487/RFC7854, June 2016, <http://www.rfc-editor.org/info/rfc7854>.

## Authors' Addresses

Yi Zheng
China Unicom
No.9, Shouti Nanlu, Haidian District
Beijing 100048
China

Email: zhengyi39@chinaunicom.cn

Xu Shiping
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
P.R. China

Email: xushiping7@huawei.com

Dhruv Dhody
Huawei Technologies
Divyashree Techno Park, Whitefield
Bangalore, Karnataka 560066
India

Email: dhruv.ietf@gmail.com